UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA

Departamento de Computación

MEMORIA DE TESIS DOCTORAL

SHIVA: UN SISTEMA HEURÍSTICO E INTEGRADO PARA LA VALIDACIÓN DE SISTEMAS INTELIGENTES

Autor: Eduardo Mosqueira Rey

Director: Vicente Moret Bonillo

A Coruña a 17 de Julio de 1998

UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA

Departamento de Computación

MEMORIA DE TESIS DOCTORAL

SHIVA: UN SISTEMA HEURÍSTICO E INTEGRADO PARA LA VALIDACIÓN DE SISTEMAS INTELIGENTES

Autor: Eduardo Mosqueira Rey

Director: Vicente Moret Bonillo

A Coruña a 17 de Julio de 1998

Vicente Moret Bonillo, Profesor Titular del Departamento de Computación de la Facultad de Informática de la Universidad de A Coruña,

CERTIFICA QUE: El proyecto titulado "SHIVA: Un Sistema Heurístico e Integrado para la Validación de Sistemas Inteligentes", ha sido realizado por D. Eduardo Mosqueira Rey bajo mi dirección, en el Departamento de Computación de la Facultad de Informática de la Universidad de A Coruña, y constituye su Tesis Doctoral.

A Coruña, a 17 de julio de 1998

Fdo.: Vicente Moret Bonillo

Director de la Tesis

A Mich, por sus "cariñosos"
mordiscos y arañazos.

AGRADECIMIENTOS

A Vicente Moret Bonillo, director de la tesis, por haber hecho posible que este trabajo sea una realidad.

A Ricardo Cao, por sus oportunos consejos referentes a la estadística.

Al Departamento de Ingeniería Biomédica y al Departamento de Pediatría del Medical College of Georgia; así como al Servicio de Cuidados Intensivos y la Unidad de Obstetricia y Ginecología del Complejo Hospitalario Juan Canalejo de A Coruña por su colaboración en el apartado de resultados.

A mis compañeros del Laboratorio LIDIA, en especial a Alfonso por sus amenos cafés.

A Pi, por facilitarme información sobre el método Delphi e ilustrarme las diferencias entre el Delphi Lockiano, Kantiano y Hegeliano (da gusto ver que hay cosas que no cambian).

A mis padres, por permitirme realizar mis deseos.

A Belén, por estar siempre conmigo.

A mis amigos de Corme y A Coruña, sobre todo aquellos que acaben de perder su libertad (aunque fuera en una vicaría).

A mi abuela Teresa, que aunque ya no está con nosotros se que nos está viendo.

Este trabajo ha sido posible gracias a la financiación de la Comisión Interministerial de Ciencia y Tecnología (CICyT) mediante proyectos de investigación, y al interés mostrado por Hewlett Packard y Sun Microsystems como entidades promotoras y observadoras de dichos proyectos.

CONTENIDO

1. INTRODUCCIÓN

El principio es la mitad de todo.
Pitágoras. (Filósofo griego. 571 – 497 a.c.).

La máquina analítica no tiene pretensión alguna de originar nada. Sólo hace lo que nosotros sabemos cómo ordenarle que haga.
Ada Byron. (Condesa de Lovelace).

El computador puede calcular mejor que ningún humano. Pero hay algo más allá del cálculo, y ese algo es nuestra comprensión sobre la naturaleza del ajedrez.
Garry Kasparov. (Ajedrecista ruso campeón del mundo. 1963 –).

Desde la antigüedad no ha dejado de transmitirse el mito de la existencia de máquinas inteligentes. Entre las primeras referencias a una actividad reveladora de la inteligencia artificial se halla la Ilíada (en la que el dios Hefaistos había construido mesas con ruedas de funcionamiento autónomo y mujeres de oro que le servían) y la tradición judía del Golem (en la que un rabino daba vida a un hombre hecho de madera o arcilla).

Sin embargo no fue hasta el siglo XVI, en el que la medicina descubrió como el funcionamiento de los órganos internos se asemejaba a distintas máquinas (el corazón a una bomba, los pulmones a un fuelle, etc.) cuando se hizo natural la idea de que podría construirse un mecanismo inteligente sin ayuda divina.

Los autómatas más logrados fueron los diseñados por Vaucanson en el siglo XVIII (un flautista que tocaba una tonadilla y un pato que nadaba, batía las alas, tragaba comida y devolvía excrementos simulados). Hacia el final de ese siglo un autor anónimo publicó el primer trabajo verdadero de inteligencia artificial: la descripción de un método automático para componer minuetos.

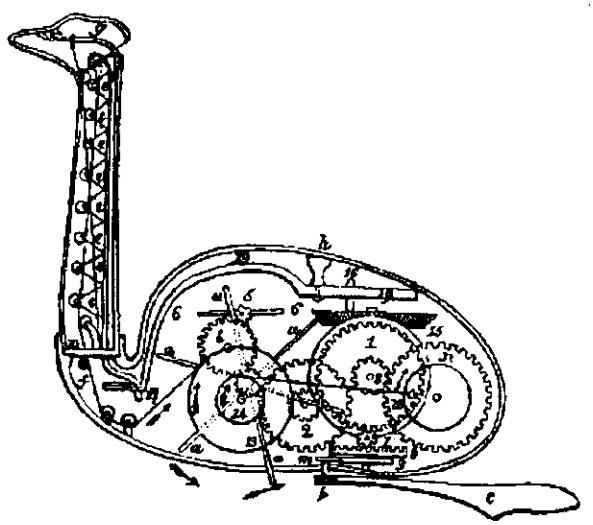


Figura 1.1. Pato de Vaucanson.

Ya en el siglo XX la inteligencia artificial se populariza a través de los escritores de ciencia ficción y surge el termino *robot*, acuñado por el escritor checo Karel Capek en 1920 y que en su idioma significa trabajo forzado (en clara referencia a que los robots eran destinados a realizar labores que el hombre no deseaba hacer).

Pero no fue hasta mediados del siglo XX cuando el mito comenzó a convertirse en realidad. El año 1956 es considerado como un punto de inflexión a raíz de una

reunión que organizó John McCarthy en el Dartmouth College y en la cual se acuña por primera vez el término *inteligencia artificial* (IA). También asistieron Marvin Minsky, Allan Newell y Herbert Simon que, junto con McCarthy, son considerados como los pioneros de la inteligencia artificial. Los asistentes confiaban en hacer progresar la cibernética estudiando los modelos del cerebro y extendiendo las posibilidades de los mecanismos de regulación, pero durante esta reunión Newell y Simon presentaron su "Logic Theorist" (un programa que podía demostrar teoremas de la lógica de proposiciones) mostrando así a los participantes como utilizar las posibilidades de manipulación de símbolos del ordenador.

De esta forma surgió una división dentro de los investigadores de la IA. Por un lado estaban aquellos que planteaban la emulación de la actividad cerebral y, en la medida de lo posible, la réplica de su estructura. Las técnicas utilizadas se denominaron técnicas conexionistas ya que pretendían emular las redes neuronales biológicas mediante redes de neuronas artificiales. Por otro lado estaba la perspectiva simbólica en la que se trataba de emular el proceso de razonamiento humano pero no así su estructura. Lo que se trataba era de desarrollar al máximo los métodos de programación para tratar los símbolos y las estructuras, entre los que cabe destacar el lenguaje LISP desarrollado por John McCarthy a partir de 1960. Para más información sobre el nacimiento de la inteligencia artificial se puede consultar (Pirat, 1991).

Los primeros esfuerzos de la inteligencia artificial se destinaron a los juegos y a la traducción del lenguaje natural. Los éxitos iniciales que se obtuvieron motivaron que Newell y Simon pronunciaran una famosa predicción en 1958: "antes de 1968 será campeón de ajedrez un programa y se demostrará un importante teorema matemático". Sin embargo esta predicción ha quedado bastante lejos de llegar a cumplirse. Esto es debido a que el entusiasmo inicial de los investigadores de IA estaba puesto en sistemas que hacían un amplio uso de la combinatoria, es decir, ensayaban muchas posibilidades sin examinar ampliamente ninguna de ellas. Tales procedimientos permitían obtener resultados mínimamente aceptables en numerosos campos, pero muy raramente hacían posibles resultados óptimos.

Así, los programas de juegos analizaban jugadas con una profundidad escasa pero analizaban una gran cantidad de variantes. De la misma forma los programas de traducción automática se basaban en la utilización de un enorme diccionario en la que se tenían en cuenta algunos elementos de gramática. Los resultados podían llegar a ser aceptables, pero nunca serían clasificados como muy buenos. Gónzalez y Dankel (1993) citan el ejemplo de una compañía aérea que a principio de los 80 anunciaba orgullosa que su flota disponía de asientos forrados en cuero con el lema "fly in leather". Las máquinas de traducción automática de la época tradujeron esta frase al español como "vuele en cueros" que, evidentemente, no tiene el mismo significado.

Si queremos aumentar el rendimiento de estos sistemas es necesario ampliar la profundidad de análisis de las jugadas en los programas de juegos y ampliar las posibles interpretaciones de las palabras (incluyendo giros locales, dobles interpretaciones, etc.) en los sistemas de traducción. Pero esto implica que se produzca una explosión combinatoria y que el número de casos a tratar sea impracticable. La única solución a este problema es la inclusión de técnicas que permitan podar el árbol de búsqueda para centrarnos sólo en aquellas opciones más prometedoras (de la misma forma que un jugador de ajedrez no analiza todas las variantes sino que desecha automáticamente aquellas que representan desventajas evidentes).

Los sistemas de inteligencia artificial aplicados a juegos han tenido siempre una gran repercusión. Existen sistemas para todo tipo de juegos: el jugador de backgammon de Berliner (1980), el jugador de damas de Samuel (1963) que incluía capacidades de aprendizaje, etc. Pero si hay un juego cuyo dominio se considera exclusivo de la inteligencia humana, debido a su complejidad y su elevado número de posibles combinaciones que pueden darse, es el juego del ajedrez.

Desde siempre se ha utilizado al ajedrez como paradigma del alcance de la lógica y el razonamiento humano. Baste citar que en 1968, en la famosa película de Stanley Kubrick y Arthur C. Clarke "2001: Una Odisea Espacial", el supercomputador HAL derrotaba al ajedrez a un resignado Frank Poole que veía su derrota como inevitable. Inicialmente la escena se había rodado con el astronauta jugando a un juego denominado "Pentominoes" que acababa de salir en aquella época y se buscaba su popularización. Sin embargo Kubrick acabó rechazándola porque quería que los espectadores percibieran cuan inteligente era HAL y la mejor forma de hacerlo era derrotando de forma irremisible al ser humano.

Otro ejemplo de la visión que se tenía sobre el ajedrez es una historia corta de Arthur C. Clarke titulada "Cuarentena" (Clarke, 1977). En ella dos supercomputadores inteligentes se asombraban ante el descubrimiento de una infección que afectaba a sus máquinas, se trataba de un problema que no podría analizarse completamente ni siquiera durante el tiempo de vida del universo. Lo más asombroso del problema era que sólo utilizaba seis operadores: Rey, Reina, Alfil, Caballo, Torre y Peón. También en la famosa serie Star Trek el capitán Kirk y Mr. Spock llegaban a jugar tres veces al ajedrez ganando Kirk las tres (demostrando así la superioridad de la intuición humana frente a los procesos eminentemente lógicos de los vulcanianos).

Desde siempre se destinó un gran esfuerzo al desarrollo de programas que jugaran al ajedrez. El primer autómata de este tipo lo construyó el Baron Kempelen en 1776 y se llamaba *Turco* (Figura 1.2). Este autómata obtuvo gran repercusión sobre todo al ganar al mismísimo Napoleón y posteriormente Maelzel (el inventor del metrónomo) lo exhibió en Estados Unidos a principios del siglo XIX. Sin embargo Turco no era más que un fraude, un experto y diminuto jugador de ajedrez se escondía dentro de sus mecanismos y se encargaba de controlar el autómata. Edgar Allan Poe publicó en 1836 un largo estudio en el que denunciaba la superchería y reflexionaba sobre las propiedades de un mecanismo capaz de jugar al ajedrez (Poe, 1983).

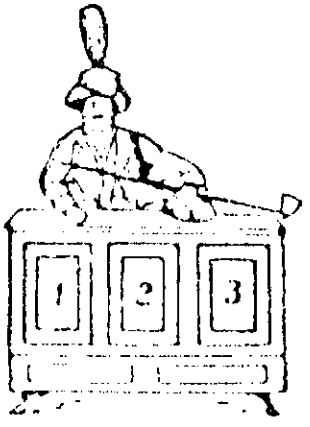


Figura 1.2 El jugador de ajedrez de Kempelen exhibido por Maelzel en Estados Unidos.

Hubo que esperar hasta el siglo XX para que se emprendieran trabajos más serios. En 1912 el español L. Torres y Quevedo construyó un autómata que jugaba las finales Rey y Torre contra Rey; en 1945 Zuse programó las reglas del juego y en 1949 Claude Shannon (promotor de la teoría de la información) indicó los principios básicos de un método al efecto. Al año siguiente Alan Turing propuso un programa de ajedrez que había simulado a mano.

Entre las primeras máquinas construidas destacan los trabajos de Alex Bernstein en el MIT entre 1958 y 1959; el MacHack de Richard Greenblatt en 1967 o los programas de Slate y Atkin: el Chess 3.0 en 1970, Chess 4.0 en 1976 y Belle en 1983, el último de los cuales alcanzó un ELO de 2552 (Hsu, 1990). Una buena descripción sobre las técnicas que emplean las computadores de ajedrez aparece en (Levy, 1986).

El computador de ajedrez más famoso hasta el momento es Deep Blue desarrollado por IBM. Su fama se debe a dos *matches* a 6 partidas que jugó con el campeón del mundo de ajedrez Garry Kasparov. En el primero Kasparov resultó vencedor, pero en el segundo (jugado en mayo de 1997) Deep Blue venció por una puntuación global de 3.5 a 2.5. De esta forma caía un viejo mito y por primera vez en la historia un computador conseguía vencer en un match a un campeón del mundo. Para obtener más información se puede consultar el documento *on-line* (IBM, 1997).

La amplia repercusión de esta derrota por los medios de comunicación hizo que la gente pensara que había llegado la época en la que, definitivamente, la máquina se había hecho superior al hombre en su tarea más preciada, el razonamiento y la inteligencia. Sin embargo esto no es así, la potencia de Deep Blue no se basa en unos procesos de razonamiento lógicos que estén al nivel de los procesos de razonamiento humano sino que se basa en una arquitectura paralela que le permite analizar unos 200 millones de posiciones por segundo y hasta 14 niveles de profundidad. Es decir, resuelve los problemas de la misma forma que hacían los primeros computadores de ajedrez, basándose en la fuerza bruta para analizar el mayor número de jugadas e incluyendo una función de evaluación (que no tendría porque ser demasiado compleja) para discriminar entre las diferentes posibilidades (baste decir que Deep Blue se ha programado en el lenguaje C, caracterizado por su rapidez pero también por estar más cercano al lenguaje máquina que al lenguaje natural). Lo que hace diferente a Deep Blue es que la tecnología actual ha permitido que el número de posiciones analizadas sea considerablemente superior a las capacidades del hombre (recordemos que un maestro suele analizar una tres jugadas por segundo).

No deberíamos vernos más alterados al ver a Deep Blue derrotar a Kasparov que al ver a una moto vencer a un corredor olímpico. Es más, el hecho de ver qué clase de tecnología es necesaria para vencer a los lentos pero eficaces procesos de razonamiento humano no hace más que ensalzar a estos últimos. Además Deep Blue siempre presenta un mismo estilo de juego (sin adaptarse a las características de su rival). El propio Kasparov ha reconocido que lo derrotó en el primer match porque pudo analizar la primera partida y descubrir su forma de jugar y sus debilidades.

Las técnicas empleadas con éxito por los programas de ajedrez no tienen por qué ser igual de adecuadas en otros dominios en los que deben predominar capacidades de razonamiento a alto nivel. Así aún existe una gran distancia entre las técnicas de reconocimiento de patrones utilizadas en el ajedrez y las utilizadas en otros aspectos de

la inteligencia. Una buena discusión sobre estos temas puede encontrarse en (Stork, 1997a) y (Stork, 1997b).

Los propios constructores de Deep Blue reconocen que su estrategia se basa fundamentalmente en la fuerza bruta y que los árboles de búsqueda alfa-beta que utiliza contrastan con los tipos de búsquedas realizadas por los grandes maestros (Figura 1.3). Es más, lo que pretenden ensalzar no es el propio programa de razonamiento de Deep Blue sino su arquitectura subyacente denominada RS/6000 a la que consideran capaz de realizar grandes cosas pero no directamente relacionadas con la inteligencia (diseño, modelado, soporte de un servidor web escalable, etc.).

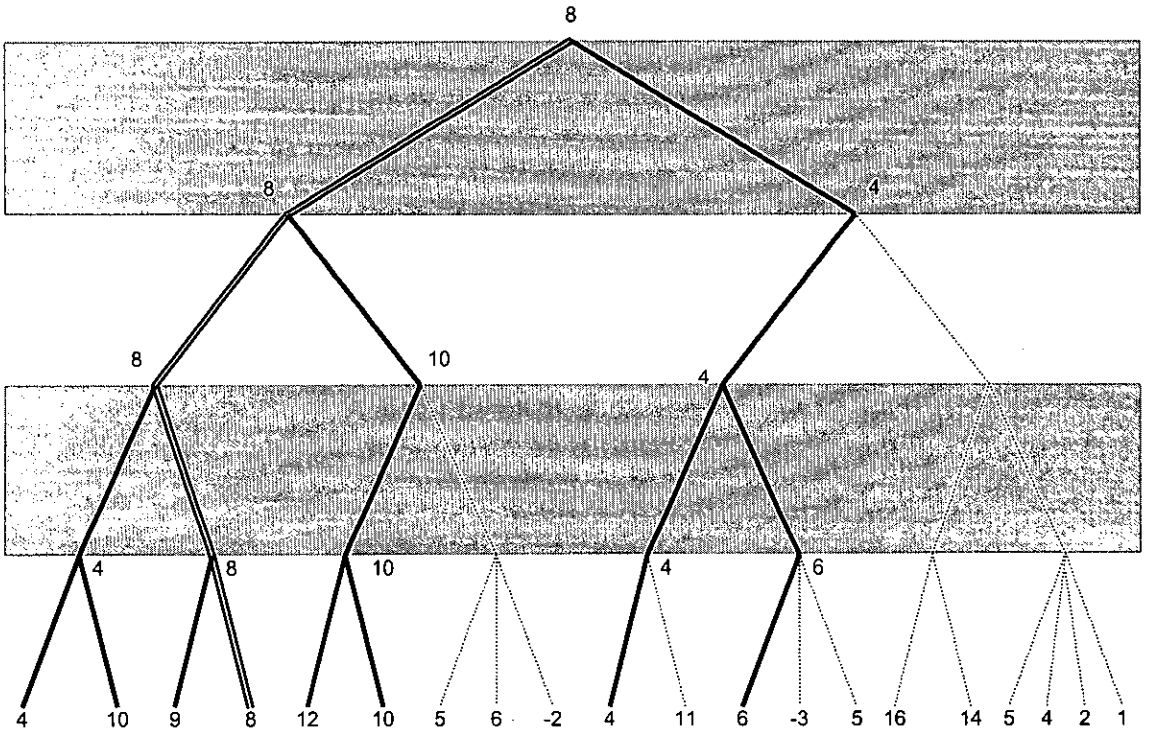


Figura 1.3. Ejemplo de árbol de búsqueda alfa-beta en donde casi la mitad del mismo (en líneas discontínuas) no ha sido necesario analizarlo para obtener la mejor jugada (suponiendo siempre las mejores respuestas por parte del contrincante). En sombreado se presentan las jugadas propias y en doble línea el análisis final resultante. Los números representan la evaluación de la posición que se obtiene tras cada jugada.

Pero este trabajo, aunque hasta ahora no se haya hablado mucho, trata sobre la validación de sistemas inteligentes, analicemos pues el rendimiento de Deep Blue. La máquina sólo ha jugado doce partidas en dos años, todas contra el mismo contrincante. En el primero de los matches sólo fue capaz de ganar una partida, para el segundo match los ingenieros de Deep Blue tenían a su disposición las partidas jugadas por Kasparov en el anterior match así como en otros campeonatos y pudieron adaptar su estrategia de forma conveniente. Kasparov no tenía ningún conocimiento de las capacidades de Deep Blue salvo por las 6 partidas anteriores. Esto tuvo una clara influencia en la estrategia seguida por Kasparov, hacer jugadas extrañas intentando salirse de posiciones conocidas almacenadas por la computadora. El problema era que estas posiciones también eran desconocidas por él. Su derrota en la última partida del segundo match fue debida a un gran error que debería haber visto pero que por cansancio pasó por alto. En palabras del subcampeón del mundo Vishwanathan Anand "IBM no quiere arriesgar la reputación de su *baby* ante cualquier máquina o mortal, por ello unos seis mil millones de personas en el mundo (exceptuando a Kasparov) tiene

más preguntas que respuestas después del match". Evidentemente los resultados de Deep Blue son prometedores (vencer a un campeón del mundo) pero hasta que la muestra utilizada en la comprobación de su rendimiento no sea más grande es imposible decir de que es capaz.

En otro tipo de juegos, como el japonés Go en donde el factor de ramificación es de 360, la estrategia de fuerza bruta no ha sido tan exitosa como en ajedrez. En estos casos se utilizan completas bases de conocimiento de reglas que propongan aquellas jugadas que pudieran ofrecer alguna posibilidad, aunque la calidad de los juegos así realizados todavía es bastante mala. De esta forma, a pesar de existir un premio de dos millones de dólares, ningún programa de Go ha conseguido derrotar a los mejores jugadores.

1.1. Definición de inteligencia artificial

> Puede llegar el día en que la inteligencia humana sea definida como aquello no factible por las máquinas.
> *Herman Kahn. (Matemático estadounidense. 1.922 –).*
>
> La tontería es infinitamente más fascinante que la inteligencia. La inteligencia tiene sus límites, la tontería no.
> *Claude Chabrol. (Cineasta francés. 1.930 –).*
>
> Inteligencia militar son dos términos contradictorios.
> *Groucho Marx. (Cómico estadounidense. 1.890 – 1.977).*

Hemos hablando de la inteligencia artificial, pero aún no la hemos definido de forma precisa. Una definición académica la definiría como "el subcampo de la ciencia de la computación que se centra en la capacidad del ordenador en manipular símbolos no numéricos e inferir nuevos hechos a partir de un conjunto conocido de hechos" (Carrico et al., 1989). Sin embargo una definición mucho más práctica sería "la IA es el estudio de ideas que permiten a los computadores ser inteligentes" (Winston, 1984).

Es muy difícil encontrar una definición que sea universalmente aceptada por todos. Russell y Norvig (1995) plantean que todas las posibles definiciones caen en cuatro categorías (Tabla 1.1). Por un lado tenemos una dimensión que distingue a las definiciones entre aquellas que hacen referencia a los procesos mentales de los sistemas o a su conducta. Por otro lado tenemos otra dimensión que hace referencia al elemento de comparación del sistema (la eficiencia humana o la racionalidad, entendiendo que un sistema es racional si hace lo correcto).

	Eficiencia humana	Racionalidad
Procesos mentales	Sistemas que piensan como humanos	Sistemas que piensan racionalmente
Conducta	Sistemas que actúan como humanos	Sistemas que actúan racionalmente

Tabla 1.1. Agrupación de las definiciones de Inteligencia Artificial según Russell y Norvig

Las definiciones que caen dentro de la categoría "sistemas que actúan como humanos" han dado lugar a uno de los más populares métodos de validación, la prueba de Turing, propuesta por el matemático Alan Turing (1950) y que básicamente consiste en definir a una máquina como inteligente si su forma de actuar es indistinguible de la forma de actuar de los humanos. Dentro de la categoría "sistemas que piensan como humanos" podemos destacar los trabajos de Newell y Simon (1961) con su solucionador

general de problemas, con el que pretendían no sólo resolver correctamente los problemas propuestos sino también seguir líneas de razonamiento similares al razonamiento humano.

Por otro lado, y dejando a un lado la comparación humana, tenemos los "sistemas que piensan racionalmente", en donde se incluye la tradición logicista por crear sistemas inteligentes y los "sistemas que actúan racionalmente" en donde incluimos a los agentes racionales (sistemas capaces de percibir y actuar de forma racional).

El problema de que existan tantas definiciones de inteligencia artificial es debido a que no existe un definición clara del término inteligencia: ¿es la habilidad de razonar?, ¿es la capacidad de adquirir conocimientos y aplicarlos? ¿o es la capacidad de percibir y manipular objetos? (Cortés et al., 1993). Todo esto entra ya dentro del campo de la filosofía por lo que intentaremos simplificar las cosas definiendo a qué se dedica la IA sin entrar más en cuestiones de si los sistemas son verdaderamente inteligentes o no. La IA se ocupa de aquellos problemas para los que no existen soluciones algorítmicas satisfactorias, o bien aquellos que sobrellevan un manejo explícito del conocimiento.

La inteligencia artificial cubre un amplio rango de áreas entre las que podemos destacar la ciencia cognitiva, la robótica, la visión artificial, la comprensión del lenguaje natural, el reconocimiento de sonidos, los sistemas basados en conocimiento, etc (Carrico et al., 1989).

1.2. Sistemas basados en conocimiento y sistemas expertos

> Un experto es uno que ha cometido todos los errores que pueden cometerse, en un campo reducido.
> *Niels Henrick Börh. (Físico danés. 1.885 – 1.962).*
>
> La experiencia es esa cosa maravillosa que te permite reconocer un error cuando vuelves a cometerlo.
> *James Jones. (Escritor estadounidense. 1.921 – 1.977).*
>
> Lo que tenemos que aprender a hacer lo aprendemos haciéndolo.
> *Aristóteles, (Filósofo griego. 384 – 322 a.c.).*

Los sistemas o programas de IA presentan cierto comportamiento inteligente a partir de la aplicación de una serie de técnicas (búsquedas heurísticas, inferencias, etc.) mediante las cuales no garantizan encontrar soluciones óptimas, pero sí permiten garantizar el hallazgo de soluciones aceptables.

Dentro de estos sistemas podemos destacar a los *sistemas basados en conocimiento*, que representan por sí mismos toda un área de estudio. Estos sistemas se definen, desde una perspectiva conceptual, como "sistemas computerizados que emulan la capacidad de resolución del ser humano, utilizando sus mismas fuentes de conocimiento en un dominio específico". Desde el punto de vista de la arquitectura la principal característica de los sistemas basados en conocimiento es que separan el conocimiento del dominio de las estructuras de control que se encargan de manejarlo.

Esta separación entre conocimiento y control es muy importante ya que las personas que desarrollan este tipo de programas (generalmente conocidas como ingenieros del conocimiento) fijan todo su esfuerzo en la construcción de la base de

conocimientos del sistema, reutilizando las estructuras de control que ya se han probado con éxito y que no tienen por que ser excesivamente complejas. En palabras del pionero de la IA Edward Feigenbaum "el poder reside en el conocimiento no en los métodos de control" (Luger y Stubblefield, 1993).

Los sistemas basados en conocimiento se subdividen a su vez en otros tipos de sistemas: sistemas expertos (SSEE), sistemas gestores basados en el conocimiento, sistemas inteligentes de desarrollo de software (ICASE), sistemas tutores inteligentes, etc (Cortés et al., 1993). De entre éstos destacaremos a los *sistemas expertos*, que son especializaciones de los sistemas basados en conocimiento que tratan de resolver problemas del mundo real, limitados en tamaño, pero intelectualmente complejos, y que habitualmente son resueltos por expertos humanos. La construcción de sistemas expertos va a requerir el empleo de las técnicas desarrolladas para construir programas de inteligencia artificial, y la utilización de las arquitecturas definidas para el desarrollo de sistemas basados en el conocimiento.

Existen programas que realizan análisis similares a los de los expertos humanos en un campo determinado utilizando su mismo conocimiento y que, sin embargo, no son clasificados como sistemas expertos. Así, por ejemplo, programas destinados a calcular los voltajes en circuitos eléctricos, esfuerzos en construcciones arquitectónicas, riesgos en los préstamos, etc. actúan en el mismo dominio que expertos humanos, de forma más eficiente y utilizando su mismo conocimiento (ecuaciones matemáticas). Sin embargo estos sistemas no presentan una separación entre conocimiento y control, y su naturaleza es típicamente algorítmica mientras que los sistemas expertos emplean conocimiento heurístico (proveniente de la experiencia).

Las principales ventajas de usar un sistema experto en un domino en particular son las siguientes:

1. *Consistencia de las respuestas.* Diferentes expertos humanos pueden presentar diferentes respuestas ante un mismo problema. El mismo experto humano también puede ofrecer resultados ligeramente diferentes en varias ocasiones. En algunos casos, estas variaciones son debidas a inconsistencias menores que no tienen ninguna, o casi ninguna, consecuencia. Sin embargo, en otros casos pueden ser debidas a errores como consecuencia del estado de salud, del estado emocional o del estrés que está sufriendo el experto. Los sistemas expertos, por otro lado, son siempre consistentes en sus respuestas ya que no se ven afectados por cuestiones emocionales o de salud que varíen su rendimiento.

2. *Reproducen un conocimiento que es escaso y permiten su amplia distribución y fácil accesibilidad.* Generalmente el conocimiento de los expertos humanos es un bien muy preciado que no poseen muchas personas dentro de cada dominio de aplicación. Con los sistemas expertos puede formalizarse este conocimiento y llevarse a cualquier parte del mundo.

3. *Son fáciles de modificar.* Al tener separado el conocimiento de la forma de manejarlo las modificaciones se pueden realizar de forma sencilla y pueden adaptarse a cambios en el dominio.

4. *Explican sus conclusiones.* Los sistemas expertos suelen incluir módulos de explicación que muestran cómo se ha llegado a una determinada conclusión. Estas explicaciones pueden utilizarse para justificar o clarificar resultados y adicionalmente para el entrenamiento de expertos en el domino.

5. *Solucionan problemas con datos incompletos.* Algunos sistemas expertos son capaces de resolver problemas para los cuales no están disponibles todos los datos o existen inexactitudes en los mismos. Esto es importante ya que en el mundo real no siempre se tiene información completa y exacta sobre un problema.

Sin embargo presentan una serie de inconvenientes como los que citamos a continuación:

1. *Las respuestas pueden no ser siempre correctas.* El conocimiento de los expertos humanos puede tener errores que pueden heredar los sistemas expertos.

2. *El conocimiento está limitado al dominio.* El conocimiento del sistema inteligente es muy específico, aunque un problema caiga fuera del rango de aplicación de su dominio el sistema siempre intentará obtener una respuesta.

3. *Los sistemas expertos carecen de sentido común.* Así un sistema inteligente puede encontrar normal que se le introduzca en la información de entrada la existencia de agua líquida a 15 grados bajo cero. Esto se podría solucionar incluyendo en el conocimiento el hecho de que el agua se hiela por debajo de cero grados a presión normal, pero es prácticamente imposible contemplar todas las posibles soluciones.

1.3. Verificación y validación de sistemas expertos

> La calidad nunca es un accidente; siempre es el resultado de un esfuerzo de la Inteligencia.
> *John Ruskin (Escritor y crítico inglés. 1.819 – 1.900).*
>
> El hombre que ha cometido un error y no lo corrige comete otro error mayor.
> *Confucio. (Filósofo chino. 551 – 479 a.c.).*

Hasta ahora hemos visto que en la definición y en las ventajas de utilización de un sistema inteligentes o un sistema experto se destacaba sobremanera su capacidad para actuar a niveles similares a los expertos humanos en determinados dominios. Pero ¿cómo podemos asegurarnos de que un sistema inteligente está actuando de forma similar a un experto humano?. La respuesta a esta pregunta la encontramos en el proceso de *verificación y validación* de sistemas expertos (conocido popularmente como V&V).

Los términos verificación y validación son ampliamente usados en la bibliografía y han aparecido múltiples definiciones de los mismos. Sin embargo lo más importante que podemos destacar de todas estas definiciones es que el principal objetivo de estos dos procesos es asegurar que los sistemas expertos ofrecen la respuesta correcta, de la forma correcta cuando se les plantea un problema determinado (Gonzalez y Dankel, 1993).

La verificación y la validación de un sistema inteligente permiten:

1. *Asegurar la calidad del producto desarrollado.* De forma que todo sistema inteligente que llegue a sus últimas fases de desarrollo cumpla unos estándares de calidad. Para ello es necesario que en la propia metodología de construcción de sistemas expertos se incluya una fase de V&V (López et al., 1990).

2. *Asegurar su utilización en dominios críticos.* Existen dominios en los cuales las decisiones que se tomen son muy importantes ya que, debido a sus consecuencias, no nos es posible reconsiderarlas. Estos dominios se denominan "dominios críticos" y la aceptabilidad de un sistema inteligente en ellos depende, en gran medida, de la realización de una fase formal de validación. Como ejemplos de dominios críticos tenemos sistemas que trabajan en UCIs, en plantas nucleares, en vehículos espaciales, etc.

3. *Asegurar su aceptabilidad en la rutina diaria.* Un sistema sólo será aceptado dentro de una rutina diaria si cumple las expectativas para las que fue construido y no comete errores. Si el sistema no tiene una fiabilidad determinada los usuarios no lo utilizarán (O'Leary, 1993).

Sin embargo la validación de sistemas expertos no constituye un campo de investigación bien estructurado. Se han desarrollado muchas aproximaciones ad-hoc al problema de la validación pero no existe una visión integral del mismo. Asimismo no existe una clasificación global de los problemas de validación ni tampoco existe una clara relación entre estos problemas y las técnicas destinadas a solucionarlos (López et al., 1990). Gupta (1993) señala que, entre los principales problemas existentes en la validación, cabe destacar: la falta de métricas de evaluación prácticas y rigurosas; la falta de especificaciones, lo que conducen a evaluaciones subjetivas; y la falta de herramientas de validación desarrolladas.

Otro problema que aparece a la hora de validar los sistemas expertos es que no existen, como en el caso del software tradicional, procedimientos que guíen el desarrollo de dichos sistemas. De esta forma la construcción de sistemas expertos aparece como un arte enigmático aplicado por unos pocos elegidos. La carencia de una metodología de desarrollo es, en parte, debida a que la inteligencia artificial es un campo de investigación relativamente reciente.

La creación de una metodología de construcción de sistemas expertos permitiría el desarrollo de este tipo de sistemas de una forma más eficiente, y con la inclusión de una fase de V&V en dicha metodología se podría comprobar mas formalmente la calidad del producto desarrollado.

1.4. Organización del trabajo

En el capítulo 2 de este trabajo se explicarán las distintas metodologías de desarrollo de sistemas convencionales existentes dentro de la ingeniería del software, y se hará una descripción de los procesos de validación que se implementan dentro de estas metodologías. Si bien las diferencias existentes entre el software convencional y los sistemas expertos impiden la aplicación directa de estos métodos, las metodologías y

modelos de validación empleados en los sistemas expertos se basan frecuentemente en aquellos provenientes de la ingeniería del software.

En el capítulo 3 nos centramos más en la ingeniería del conocimiento y en las diferencias que presenta con la ingeniería del software, exponemos las metodologías de desarrollo de sistemas expertos más comunes y cómo se estructuran los procesos de validación dentro de estas metodologías. También se describen otras fases (además de la validación y la verificación) que forman parte del proceso global de análisis de comportamiento de los sistemas expertos.

En los siguientes capítulos ya entramos más de lleno en lo que es la V&V. Así el capítulo 4 trata sobre la verificación de sistemas expertos, centrándonos básicamente en la verificación de la base de conocimientos. Se incluyen los distintos tipos de verificación existentes y las distintas herramientas desarrolladas. El capítulo 5 trata sobre la validación de sistemas expertos, centrándose básicamente en una validación orientada a los resultados y analizando los principales paradigmas a tener en cuenta. Los principales métodos estadísticos que hemos empleado en la validación se describen con detalle en el capítulo 6.

En el capítulo 7 se desarrolla una nueva metodología de validación de sistemas expertos en la que se combinan distintas técnicas estadísticas para intentar determinar la calidad del sistema desarrollado. En dicha metodología se puede ver como las características del dominio, del sistema o de la fase de desarrollo pueden afectar a la selección de las herramientas estadísticas que son más adecuadas para llevar a cabo la validación.

En el capitulo 8 se describe la herramienta SHIVA (Sistema Heurístico e Integrado de VAlidación) que permite aplicar la metodología del capítulo 7 de forma sencilla y directa. Esta metodología ha sido aplicada con éxito en diversos sistemas como se muestra en el capítulo 9.

El capítulo 10 comprende la discusión del trabajo, las conclusiones a las que hemos llegado y las principales aportaciones que se han producido, así como una breve descripción de lo que pretende ser el trabajo futuro. Finalmente incluimos varios apéndices con distinta información útil y describimos las referencias bibliográficas utilizadas en el texto.

1.5. Resumen

En este capítulo hemos tratado de exponer una perspectiva histórica de la inteligencia artificial, desde sus mitos y leyendas iniciales, pasando por los autómatas que tanto fascinaron a las gentes del siglo XVIII, hasta llegar al desarrollo actual de este campo, cuyo comienzo podemos fijar en 1956. En esta época se gestó lo que podríamos llamar el "mito del ajedrez" y que consiste en utilizar al juego del ajedrez como paradigma de la inteligencia humana.

La inteligencia artificial ha sido definida de múltiples formas diferentes aunque Russell y Norvig (1995) han establecido una clasificación de las mismas según el elemento de comparación del sistema (la eficiencia humana o la racionalidad) o a la parte del sistema que se considera inteligente (la conducta o el pensamiento).

Dentro de la inteligencia artificial podemos destacar el desarrollo de los sistemas basados en conocimiento, y mas concretamente, los sistemas expertos (que son especializaciones de los sistemas basados en conocimiento que tratan de resolver problemas del mundo real, limitados en tamaño, pero intelectualmente complejos, y que habitualmente son resueltos por expertos humanos).

Para asegurar la calidad de los sistemas expertos y permitir su utilización en la rutina diaria (especialmente si se trata de dominios críticos) estos sistemas deben ser validados de forma rigurosa a través de las fases de verificación y validación.

Al final de este capítulo se realiza un esquema de los principales puntos que forman este trabajo.

2. INGENIERÍA DEL SOFTWARE

> La programación, hoy en día, es una carrera entre los ingenieros del software, que se esfuerzan en constituir más grandes y mejores programas a prueba de idiotas, y el universo, que trata de producir más grandes y mejores idiotas. Por ahora, el universo está ganando.
> *Rich Cook.*
>
> Nunca hay tiempo para hacerlo de forma adecuada, pero siempre hay tiempo para rehacerlo completamente.
> *Ley de Iversen.*

La *Ingeniería del Software* es una disciplina dentro de la computación que se ocupa de desarrollar métodos y técnicas para definir, construir y mantener software de calidad (Mayrhauser, 1990). Existen otras muchas definiciones resaltando aspectos como el estudio de principios y metodologías (Zelkowitz, 78), la inclusión de la documentación en los desarrollos (Boehm, 1976), o incidir en el aspecto práctico del software desde dos puntos de vista: hace lo deseado y es económicamente factible (Bauer, 1972). Todas estas definiciones tienen en común el hecho de incidir en la importancia de una disciplina de ingeniería para el desarrollo del software.

La ingeniería del software nació como respuesta a lo que se dio en llamar la *crisis del software*, frase melodramática que hace referencia a la gran cantidad de problemas asociados a los métodos de desarrollo del software convencional.

Un elemento fundamental dentro de la ingeniería del software es la construcción de modelos que permitan guiar el desarrollo de los sistemas, desde las primeras etapas de análisis hasta su implementación final, a través de una serie de pasos o fases bien definidas y que van indicando a cada momento que es lo que hay que hacer. Existen muchos modelos de desarrollo de programas, también conocidos como *ciclos de vida*. A continuación veremos una breve descripción de los más utilizados.

2.1. Modelo "codifica y corrige"

Los primeros desarrollos de software seguían una "metodología" denominada *codifica y corrige* (code & fix) y que contenía dos pasos fundamentales (Boehm, 1988):

a) Escribir una parte del código.

b) Corregir los problemas encontrados en el código.

es decir, el programador empezaba codificando y ya se preocuparía más tarde de los requisitos, el diseño, el test o el mantenimiento. Este modelo se representa en la Figura 2.1.

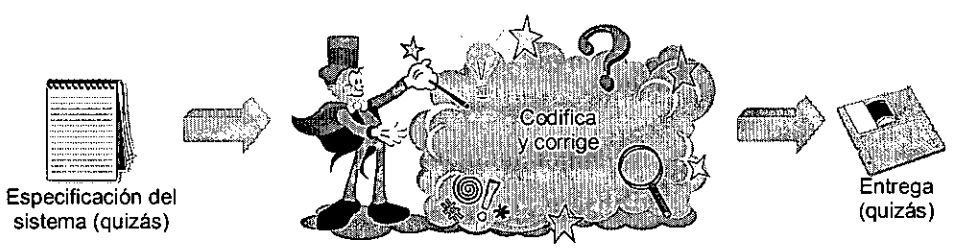


Figura 2.1. Esquema del modelo "codifica y corrige" (code & fix).

Entre las ventajas que presenta el modelo destacamos las siguientes (McConnell, 1996):

a) No conlleva ninguna gestión: no se pierde tiempo en la planificación, en la documentación, en el control de calidad, en el cumplimiento de los estándares, o en cualquier otra actividad que no sea la codificación pura.

b) Como se pasa directamente a codificar, se pueden mostrar inmediatamente indicios de progreso.

c) Requiere poca experiencia. Cualquier persona que haya escrito alguna vez un programa de ordenador está familiarizada con el modelo y puede utilizarlo.

Evidentemente este modelo presenta muchos problemas e inconvenientes, como por ejemplo:

a) Después de una serie de correcciones el código está poco estructurado, de tal forma que las subsecuentes modificaciones son muy costosas.

b) Muchas veces el software es desarrollado de forma adecuada pero no responde a las necesidades de los usuarios. Esto provoca que el proyecto sea rechazado o rediseñado de una forma muy costosa.

c) El código es complicado de corregir debido a una escasa preparación para el test y el mantenimiento.

En la Tabla 2.1 podemos ver de forma resumida estas ventajas e inconvenientes.

Ventajas	Inconvenientes
No conlleva gestión	Poca estructuración del código
Progresa inmediatamente	Diseños poco adecuados
Requiere poca experiencia	Complicado mantenimiento

Tabla 2.1. Ventajas e inconvenientes del método "codifica y corrige".

Este modelo es adecuado para pequeños proyectos pero puede resultar peligroso. Así, puede que evitemos tareas de gestión pero al basarse exclusivamente en la codificación, si se encuentran errores graves de diseño cuando la codificación está muy avanzada, su solución es muy costosa. Por ello se deduce que una forma de actuar más correcta sería postergar la codificación hasta que se halla realizado un análisis y un diseño detallado del problema a resolver. Por ello los modelos de desarrollo del software dividen la construcción de los programas en una serie de fases sucesivas.

2.2. Modelo en cascada

En este nuevo modelo de fases la codificación no es el núcleo central del desarrollo sino una fase más del proceso. Entre los modelos de fases, el más popular y el que más aceptación tiene es el *modelo en cascada* también conocido como "ciclo de vida clásico".

Según el modelo en cascada la construcción del software se hace de forma secuencial a través de una serie de fases que permiten bucles de retroalimentación de tal forma que se minimice el coste de corrección de un error determinado. Este modelo se representa en la Figura 2.2.

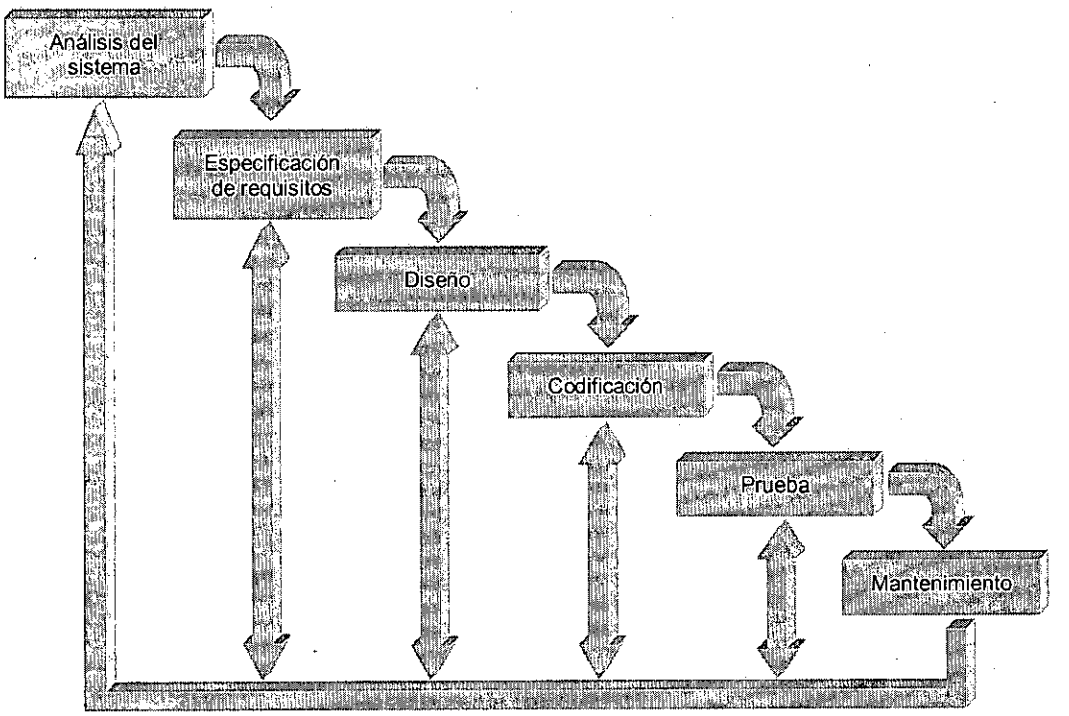


Figura 2.2. Modelo en cascada de desarrollo de software.

Las fases de las que consta el modelo son las siguientes:

1. *Análisis del sistema*: Se realizan estudios para comprobar la viabilidad del sistema y estudios de coste-beneficio.

2. *Especificación de requisitos*: Se genera un documento en el que se describen todas las metas que se quieren alcanzar y las características del sistema a construir.

3. *Diseño*: Es una fase crítica dentro del modelo. Un sistema perfectamente diseñado será relativamente fácil de implementar, validar y mantener. El diseño se divide en diseño preliminar y diseño detallado. En el *diseño preliminar* se determina la estructura de alto nivel del software, incluyendo diagramas de flujo de datos, gráficos estructurales, y la posibilidad de determinación del lenguaje de implementación. En el *diseño detallado*, por otro lado, se especifican detalles de bajo nivel incluyendo los diagramas de flujo de los módulos definidos en los gráficos estructurales y las estructuras de datos a utilizar. En definitiva, el diseño pretende traducir los requisitos en una representación del software que pueda ser establecida de forma que obtenga la calidad requerida antes de que comience la codificación.

4. *Codificación*: El diseño debe traducirse en una forma legible para la máquina. Esta etapa incluye la escritura y el depurado del código para cada modulo, la integración de estos módulos y la comunicación del sistema con componentes externos.

5. *Prueba*: Una vez que se ha construido el sistema hay que comprobar su funcionamiento. Estas pruebas incluyen verificar si se cumplen los requisitos iniciales y si el sistema obtiene las soluciones adecuadas al problema planteado.

6. *Mantenimiento*: Esta fase incluye todas las modificaciones que se realicen en el software después de su desarrollo. Los cambios ocurrirán debido a que se han encontrado errores, debido que el software debe adaptarse por cambios del entorno externo o debido a que el cliente requiere aumentos funcionales o del rendimiento. Esta fase es la más costosa del modelo incluyendo entre un 30% y un 80% del esfuerzo total del proyecto (Shooman, 1983). Según Pressman (1998) el coste del mantenimiento ha crecido rápidamente durante los últimos 20 años debido a lo cambiante de los sistemas informáticos. Sólo siguiendo una metodología formal se puede minimizar el impacto del mantenimiento en un sistema.

El modelo en cascada es el ciclo de vida más usado dentro de la ingeniería del software. Permite realizar el progreso del proyecto a través de una secuencia ordenada de pasos realizando una revisión al final de cada etapa para determinar si se está preparado para pasar a la siguiente. De esta forma los errores se encuentran antes de que su corrección sea demasiado costosa. Sin embargo con el paso de los años han aparecido una serie de críticas que podemos resumir en los siguientes puntos (Pressman, 1998):

1. Los proyectos reales raramente siguen el flujo secuencial que propone el modelo. Siempre ocurren iteraciones y se crean problemas en la aplicación del paradigma.

2. El modelo presupone que el cliente podrá establecer explícitamente al principio todos los requisitos. Sin embargo, lo normal es que esto no sea cierto y que aparezcan incertidumbres que son difíciles de acomodar

3. El cliente debe tener paciencia. Una versión funcionando del programa no estará disponible hasta las etapas finales del desarrollo del proyecto. El coste de un error importante no detectado hasta que el programa esté funcionando puede ser muy elevado.

Un resumen de las ventajas e inconvenientes del modelo en cascada puede verse en la Tabla 2.2.

Ventajas	Inconvenientes
Realización del proyecto de forma ordenada	Complicaciones si el flujo no es secuencial
Permite detectar errores importantes en las primeras fases del desarrollo	Dificultades en establecer todos los requisitos en la primeras fases
	Resultados visibles sólo en las etapas finales

Tabla 2.2. Ventajas e inconvenientes del modelo en cascada.

Debido a estas características el modelo en cascada es apropiado en aquellos dominios en los que los errores en las primeras fases de análisis de requisitos y diseño

son improbables. Estos dominios son generalmente aquellos sobre los que se tiene una amplia información, es decir, dominios bien conocidos (Lowry y Duran, 1989).

2.3. Modelo de prototipos

Una posible solución a los problemas expuestos para el modelo en cascada pasa por la construcción de *prototipos*. El desarrollo del prototipo es un proceso que facilita al programador la creación de un modelo de software a construir. Los prototipos pueden ser de tres formas (Pressman, 1998): *prototipo en papel*, describiendo las interacciones hombre-máquina de forma que facilite al usuario la comprensión de cómo se producirá tal interacción; *prototipo funcional,* que implementa algunos subconjuntos de la función requerida al software deseado; o un *prototipo ya existente*, que no es más que un programa que ya existe y que ejecuta parte o toda la función deseada, pero que tenga otras características que deban ser mejoradas en el nuevo trabajo de desarrollo.

Botting (1985) distingue también otros dos tipos de prototipos: las *maquetas* (mock-ups) que muestran sólo el interfaz de usuario de forma que el usuario vea la forma en la que se producirá la interacción con el programa; y los *paneles de pruebas* (breadboards) en donde se implementan una serie de funciones del producto software pero no su interfaz y suelen ser construidos por los desarrolladores para comprobar la implementación de dichas funciones.

Idealmente el prototipo sirve como mecanismo para identificar los requisitos del software. El desarrollo de un programa mediante esta técnica seguirá las siguientes fases (Amescua et al., 1995):

1. *Análisis inicial.* Analizamos el problema en términos globales, con el objetivo de conseguir una base sólida para los siguientes pasos.
2. *Diseño y realización.* Se construye la primera versión del prototipo.
3. *Evaluación.* Los usuarios finales manejan y comprueban el prototipo.
4. *Modificación.* El prototipo es modificado hasta que los usuarios están satisfechos con él.
5. *Definición de requisitos del sistema.* Después del análisis del prototipo se pueden definir de forma casi completa los requisitos del sistema.

Estos pasos los podemos ver de forma gráfica en la Figura 2.3.

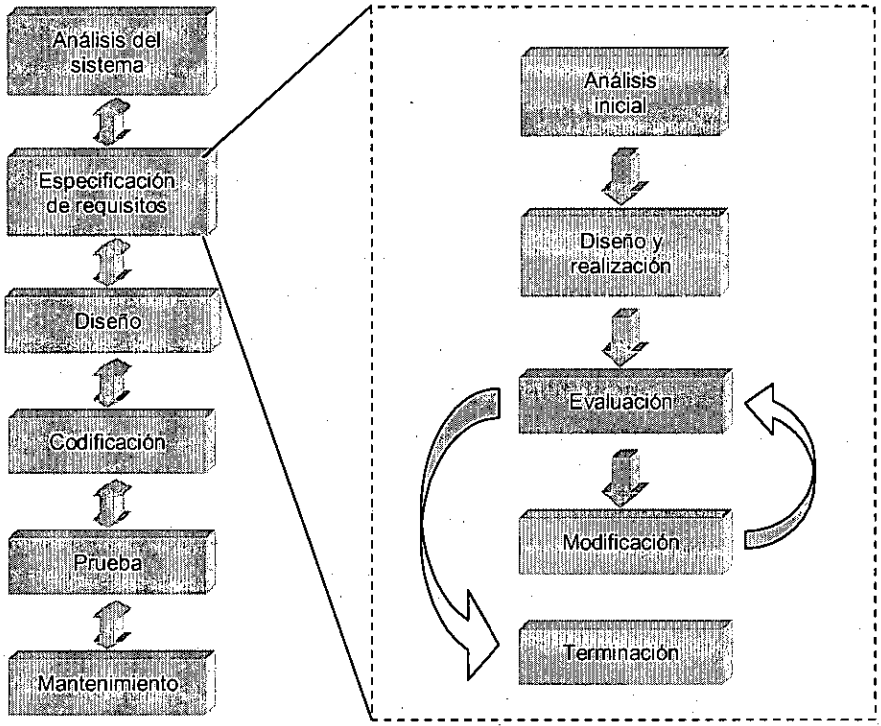


Figura 2.3. Modelo de prototipos para el desarrollo de software.

En el modelo del prototipado el usuario tiene una labor más activa en el desarrollo del sistema. Sin embargo esta técnica no está exenta de problemas (Pressman, 1998):

1. El usuario ve funcionando lo que parece ser una versión del software ignorando que el prototipo se ha hecho sin tener en cuenta aspectos de calidad o mantenimiento a largo plazo. Generalmente después de la presentación al usuario, si este está satisfecho, pide que se le apliquen "las pequeñas mejoras" al prototipo para convertirlo en el producto final, pero no entiende que es necesaria un reconstrucción completa del mismo para asegurar su buen funcionamiento a largo plazo.

2. El técnico de desarrollo realiza frecuentemente ciertos compromisos de implementación para a obtener un prototipo que funcione rápidamente. Así, se pueden utilizar herramientas inapropiadas simplemente porque están disponibles o porque se conocen. Finalmente el proyecto final se desarrolla con estas herramientas no ideales que una previa planificación hubiera rechazado.

Las ventajas e inconvenientes de los prototipos pueden verse en la Tabla 2.3.

Ventajas	Inconvenientes
Sigue un diseño estructurado como el modelo en cascada	Puede crear la falsa ilusión de que el proyecto está casi finalizado
Permite clarificar los requisitos en etapas tempranas del proyecto	Las herramientas utilizadas para el diseño rápido del prototipo pueden comprometer el posterior desarrollo del proyecto

Tabla 2.3. Ventajas e inconvenientes del modelo de prototipos.

Una clave importante para la aceptación de los prototipos está en que usuario e ingeniero del software deben acordar que el prototipo es sólo un mecanismo de definición de requisitos que, posteriormente, ha de ser descartado (al menos en parte) para construir el software real con los ojos puestos en la calidad y en el mantenimiento.

La técnica del prototipado es útil cuando los requisitos cambian con rapidez, cuando el usuario es incapaz de definir de forma detallada los requisitos iniciales, o cuando los desarrolladores no están seguros de la arquitectura o los algoritmos adecuados a utilizar.

2.4. Modelo incremental

El modelo de prototipos sigue siendo el modelo en cascada pero con variaciones a la hora de establecer los requisitos. Barry Boehm (1981), entre otros autores, sugirió que el desarrollo del software podía ser realizado de forma incremental combinando elementos del modelo en cascada y del modelo de prototipos (Figura 2.4).

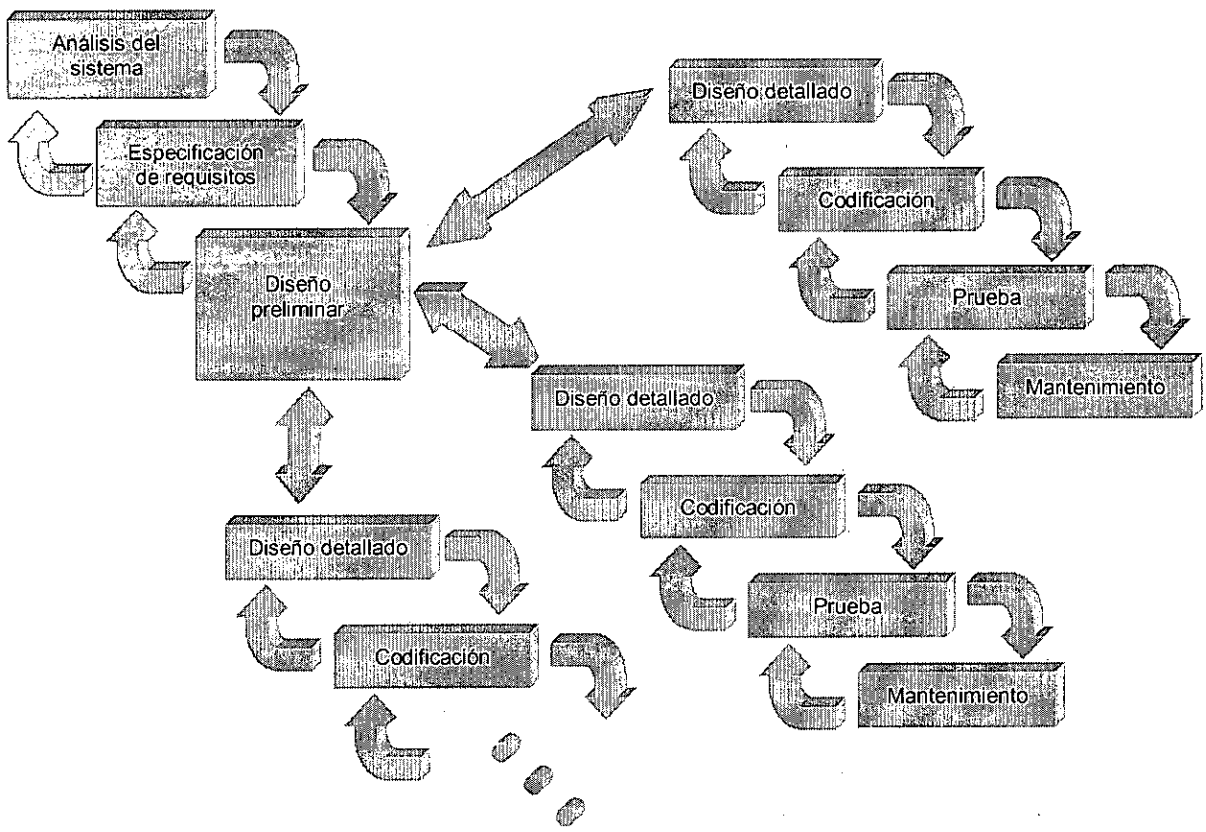


Figura 2.4. Modelo incremental de desarrollo de software.

En esta aproximación se comienza el desarrollo del sistema para satisfacer un subconjunto de requisitos especificados. Las siguientes versiones, que pueden desarrollarse en paralelo, implementarán los requisitos que faltan. De esta forma podemos ir construyendo las partes más elementales del sistema (y validarlas con los usuarios) mientras dejamos las partes más desconocidas o complicadas para posteriores diseños incrementales. A diferencia del modelo de prototipos, en el modelo incremental cada desarrollo es un producto operacional .

Las ventajas de este modelo son las siguientes:

1. Se logra una pronta disponibilidad del sistema que, aunque todavía incompleto, es por lo menos utilizable y satisface algunas de las necesidades básicas de información.

2. Los sucesivos desarrollos incrementales son más fáciles de validar que un producto global.

3. Permite incluir la experiencia de los usuarios para redefinir el producto de una forma menos cara que si tuviéramos que hacer un rediseño total.

El inconveniente de esta técnica es que no todos los sistemas pueden subdividirse en incrementos funcionales y, en caso de ser posible dicha subdivisión, pueden aparecer interdependencias imprevistas por los desarrolladores. Además la validación separada de los distintos desarrollos incrementales no asegura que el funcionamiento del sistema global sea correcto. Un resumen de las ventajas e inconvenientes de este método puede verse en la Tabla 2.4.

Ventajas	Inconvenientes
Pronta disponibilidad del sistema	No todos los sistemas pueden subdividirse en incrementos funcionales
Facilidades en la validación de los desarrollos incrementales	Las interdependencias imprevistas entre los diferentes incrementos funcionales pueden complicar el desarrollo
El proyecto se puede redefinir después de cada desarrollo	La validación separada de los desarrollos incrementales no asegura una validación global correcta
Los distintos desarrollos se pueden realizar en paralelo	

Tabla 2.4. Ventajas e inconvenientes del modelo incremental.

El modelo incremental es adecuado para aquellos proyectos que se pueden subdividir fácilmente en módulos funcionales o que su desarrollo es necesario hacerlo en distintas etapas de complejidad creciente.

2.5. Modelo evolutivo

El modelo evolutivo (McCracken y Jackson, 1982) hace énfasis en lograr un sistema flexible que se pueda expandir de tal manera que se pueda realizar muy rápidamente una versión modificada del mismo cuando los requisitos cambien. Sucede una situación parecida cuando se realiza un nuevo prototipo para ajustarse a los cambios de los requisitos de los usuarios.

El modelo evolutivo se diferencia del prototipado clásico en el concepto que tiene sobre los requisitos de diseño. En el prototipado se supone que estos requisitos existen y se utilizan técnicas de refinamiento para establecerlos. En el modelo evolutivo se supone que los requisitos son cambiantes a corto plazo y se pretende diseñar un sistema que sea rápidamente reemplazable por uno nuevo cuando se produzca un cambio en dichos requisitos.

El modelo evolutivo también se diferencia del modelo incremental en que en el primero se desarrolla una nueva versión de todo el sistema, mientras que en el segundo se parte de una versión previa sin cambios, más un número de nuevas funciones. Por supuesto, es posible desarrollar un sistema de forma incremental y al mismo tiempo seguir una aproximación evolutiva para desarrollar componentes particulares del mismo (Amescua et al., 1995).

Este modelo es apropiado para aquellos dominios poco conocidos o en sistemas altamente interactivos (Lowry y Duran, 1989). Un representación del mismo sería la Figura 2.5.

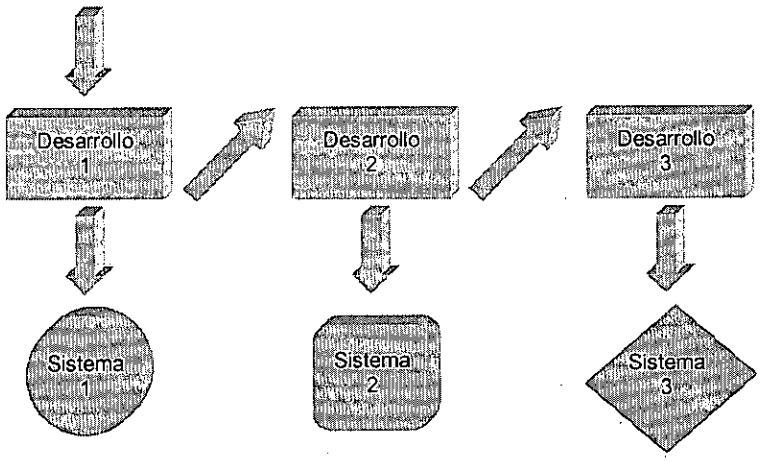


Figura 2.5. Esquema del modelo evolutivo.

Sin embargo el modelo evolutivo también tiene sus dificultades (Boehm, 1989):

1. Generalmente es difícil de distinguir del viejo modelo de "codifica y corrige" cuyo código "espagueti" y su falta de planificación fueron las motivaciones para la construcción de modelos de desarrollo del software.

2. Está basado en la asunción no realista de que el sistema será lo suficientemente flexible para acomodar los caminos evolutivos no planificados. Además a medida que el sistema se vaya desarrollando más difícil será su modificación.

Las ventajas e inconvenientes de este modelo se resumen en la Tabla 2.5.

Ventajas	Inconvenientes
Fácil adaptación a requisitos cambiantes	Puede llevarnos de nuevo a la técnica "codifica y corrige"
Produce un sistema completamente funcional en cada iteración	El sistema no siempre será los suficientemente flexible para acomodar caminos evolutivos no planeados

Tabla 2.5. Ventajas e inconvenientes del modelo evolutivo.

2.6. Modelo en espiral

Este modelo fue desarrollado por Boehm en 1988 en un intento de aunar las ventajas de los modelos anteriormente vistos (Boehm, 1988). Así el modelo en espiral

incluye validaciones tempranas del producto a través de prototipos y también un desarrollo incremental del software.

El modelo en espiral consiste en la repetición cíclica de una serie de pasos (Figura 2.6). Estos pasos cambian ligeramente según en la etapa de desarrollo en la que nos encontremos, pero siguen una estructura similar a la del modelo en cascada. También incluye una nueva función que no está considerada en otras metodologías como es el análisis de riesgos.

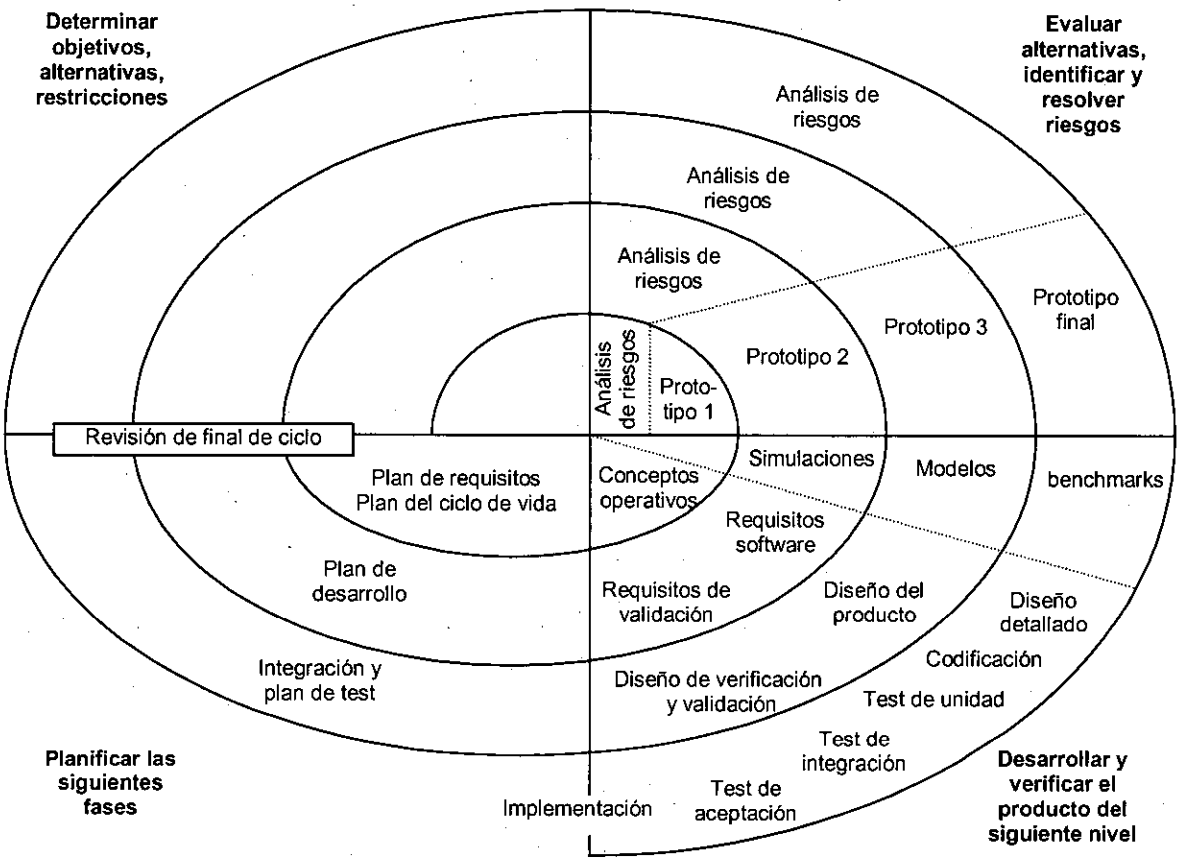


Figura 2.6. Esquema del modelo espiral.

En este modelo la dimensión radial representa el coste acumulado incurrido al realizar los distintos pasos, mientras que la dimensión angular representa el progreso realizado en cada ciclo en particular. Los ciclos del modelo se dividen en las siguientes fases:

1. *Identificación.* Determina los objetivos del ciclo, las diferentes alternativas que pueden utilizarse para cumplir los objetivos y que restricciones existen en estas alternativas.

2. *Evaluación.* Examina los objetivos y las restricciones impuestas en las alternativas para descubrir los riesgos implicados.

3. *Formulación.* Desarrolla una estrategia que resuelve las incertidumbres y los riesgos. Puede incluir el desarrollo de cuestionarios, pruebas "benchmark", simulaciones y/o prototipos.

4. *Desarrollo del producto*. Este paso dentro del ciclo depende de los riesgos remanentes que todavía existen. Si estos riesgos afectan al rendimiento del sistema o a aspectos relativos al interfaz de usuario se sigue un desarrollo evolutivo. Sin embargo, si los riesgos existentes hacen referencia al desarrollo del programa o al control del interfaz, se sigue una aproximación en cascada, modificada de forma apropiada para incluir el desarrollo incremental.

5. *Validación*. Se valida el sistema construido y se planifican lo que serán las subsecuentes fases.

6. *Revisión de final de ciclo*. En la que se desarrolla una revisión que cubre todos los productos desarrollados durante el ciclo anterior, incluidos los planes para el siguiente ciclo y los recursos requeridos para llevarlos a cabo. El principal objetivo de esta revisión es asegurarse de que todas las partes están comprometidas con la aproximación para la siguiente fase.

Sin embargo, esta división del modelo no debe tomarse de forma literal. No es importante que la espiral tenga cuatro ciclos, y no es importante tampoco que se realicen exactamente las seis fases como se indica, aunque se trata de un orden apropiado a utilizar. Las iteraciones de la espiral se pueden adaptar a las demandas de cada proyecto en concreto.

Las ventajas de este modelo son las siguientes (Pressman, 1998):

1. Mantiene un enfoque sistemático correspondiente a los pasos sugeridos por el ciclo de vida clásico, pero incorporándolo dentro de un marco de trabajo interactivo que refleja de forma más realista el mundo real.

2. Utiliza la creación de prototipos como un mecanismo de reducción de riesgos pero, lo que es más importante, permite a quién lo desarrolla aplicar el enfoque de creación de prototipos en cualquier etapa de la evolución del producto.

3. Demanda una consideración directa de riesgos técnicos en todas las etapas del proyecto y, si se aplica adecuadamente, debe reducir los riesgos antes de que se conviertan en problemáticos.

4. Es adecuado para sistemas software grandes, complejos y/o ambiciosos.

Sin embargo el modelo en espiral también presenta una serie de inconvenientes:

1. Requiere una considerable habilidad para la valoración del riesgo, y cuenta con esta habilidad para el éxito. Si no se descubre un error a tiempo, indudablemente surgirán problemas.

2. Se trata de un modelo complicado que requiere una gestión concienzuda, atenta y conocimientos profundos. Puede ser difícil convencer a grandes clientes (particularmente en situaciones bajo contrato) que el enfoque evolutivo es controlable.

3. El modelo en sí mismo es relativamente nuevo y no se ha usado tanto como el modelo en cascada o la creación de prototipos. Pasarán unos cuantos años antes de que se pueda determinar con absoluta certeza la eficacia de este importante nuevo paradigma.

Las ventajas y los inconvenientes del modelo se resumen en la Tabla 2.6.

Ventajas	Inconvenientes
Sigue un esquema de desarrollo incremental que refleja de forma realista el mundo real	Requiere una considerable habilidad para la valoración del riesgo
Permite la creación de prototipos en cualquier etapa de la evolución del producto.	Se trata de un modelo complicado que requiere una gestión concienzuda, atenta y conocimientos profundos
Reduce los riesgos del proyecto antes de que se conviertan en problemáticos	Es un modelo relativamente nuevo y no se ha usado tanto como el modelo en cascada o la creación de prototipos
Es adecuado para sistemas software grandes, complejos y/o ambiciosos	

Tabla 2.6. Ventajas e inconvenientes del modelo incremental.

2.7. Nuevas técnicas en la ingeniería del software

Los distintos modelos de construcción del software vistos en los apartados anteriores pueden mejorarse a partir de la utilización de nuevas técnicas de ingeniería del software. Dentro de estás técnicas podemos destacar las siguientes:

a) *Reutilización de componentes*. Consiste en establecer bibliotecas de componentes software que estén disponibles para los desarrolladores y que permitan su reutilización cuando sea necesario.

b) *Técnicas de cuarta generación*. Se orientan hacia la posibilidad de especificar el software a un nivel más próximo al lenguaje natural y de automatizar los procesos de la ingeniería del software a través de herramientas CASE (Computer Aided Software Engineering).

c) *Métodos formales*. Utilizan la base matemática para describir los sistemas de computadoras. Dentro de estos podemos destacar metodologías como el "cleanroom" o "sala limpia" que se basa en la prevención de errores en vez de en su corrección. Esta metodología utiliza la teoría matemática para el desarrollo y la teoría estadística para las pruebas (Simon and Herz, 1998). Presenta los problemas de consumir mucho tiempo, requerir de los ingenieros del software un conocimiento de los métodos formales y no ser fácil la comunicación de las especificaciones o del diseño a un cliente no preparado (Sobey, 1998)

Para una discusión en profundidad de la ingeniería del software se puede consultar (Pressman, 1998), (McConnell, 1996), (Jones, 1990), (Shooman, 1983) o (Macro y Burton, 1987).

2.8. Verificación y validación en sistemas convencionales

Como hemos podido ver a la hora de describir las metodologías de construcción de sistemas convencionales en todas ellas se incluyen fases para llevar a cabo la verificación y la validación del software. Esta verificación y esta validación se lleva a cabo a través de una estrategia de prueba del software que describe los pasos a llevar a cabo como parte de la prueba; cuándo se deben planificar y realizar esos pasos y cuánto esfuerzo, tiempo y recursos serán requeridos.

La verificación y la validación se basan en lo que se denominan pruebas del software, que no son más que un conjunto de actividades que se pueden planificar por adelantado y llevar a cabo sistemáticamente. Las pruebas difieren según el punto en que nos encontremos y son realizadas por el desarrollador del software y, en grandes proyectos, por grupos de prueba independientes.

Pero qué es la *verificación* y la *validación*. La verificación se puede definir como el conjunto de actividades que aseguran que el software implementa correctamente una función específica, mientras que la validación se definiría como el conjunto de actividades que aseguran que el software construido se ajusta a los requisitos del cliente. Boehm (1981) lo establece de otra forma:

Verificación: ¿Estamos construyendo el producto correctamente?

Validación: ¿Estamos construyendo el producto correcto?

La estrategia de prueba del software convencional consta de cuatro pasos que se llevan a cabo secuencialmente. Estos pasos se representan en la Figura 2.7.

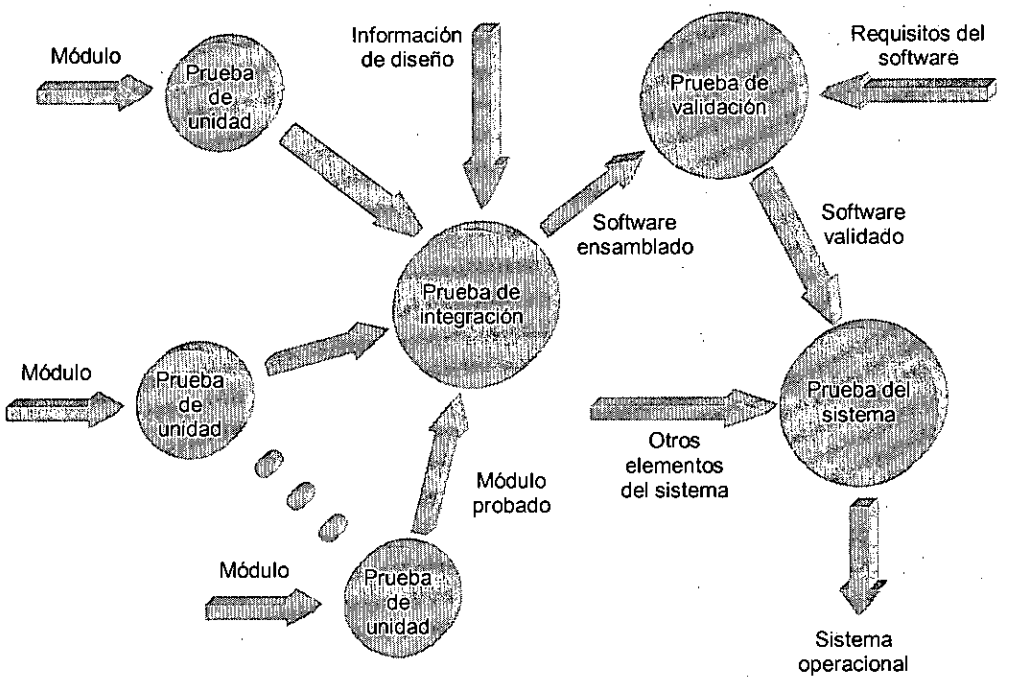


Figura 2.7. Pasos de prueba del software según Pressman (1988).

Inicialmente la prueba se centra en cada módulo individual, asegurando que funcionan adecuadamente como una unidad (*prueba de unidad*). A continuación se deben ensamblar o integrar los módulos para formar el paquete de software completo.

La *prueba de integración* se dirige a todos los aspectos asociados con el doble problema de verificación y construcción del programa. Finalmente a través de la *prueba de validación* se comprueba si el software satisface todos los requisitos funcionales y de rendimiento. La prueba restante queda fuera de los límites de la ingeniería del software entrando en el más amplio contexto de la ingeniería de sistemas computacionales. Así la *prueba del sistema* se encarga de comprobar que cada elemento del sistema global (software, hardware, bases de datos, usuarios) encajan de forma adecuada y que se alcanza la funcionalidad y el rendimiento del sistema total.

2.8.1. Prueba de unidad

La prueba de unidad centra el proceso de verificación en la menor unidad del diseño del software, el módulo. Usando la descripción del diseño detallado como guía, se prueban los caminos de control importantes con el fin de descubrir errores dentro del ámbito del módulo.

Las pruebas de unidad son siempre pruebas de la *caja blanca*. Estas pruebas se basan en comprobar que la operación interna del módulo se ajusta a las especificaciones y que todos los componentes internos se han comprobado de forma adecuada, es decir, se basan en el minucioso examen de los detalles procedimentales.

Así en la prueba de unidad se examinan las estructuras de datos locales, las condiciones límite de los bucles, los caminos básicos de las estructuras de control, etc.

2.8.2. Prueba de integración

Aunque hallamos probado todos los módulos por separado en la prueba de unidad, esto no quiere decir que cuando los integremos en el sistema su funcionamiento seguirá siendo el correcto (algunos módulos pueden tener efectos no deseados sobre otros, las estructuras de datos globales pueden presentar problemas, la imprecisión aceptada individualmente puede crecer hasta niveles inaceptables, etc).

La prueba de integración es una técnica sistemática para construir la estructura del programa mientras que al mismo tiempo se llevan a cabo pruebas para detectar errores asociados con la interacción.

La integración de módulos se suele hacer de forma incremental de forma que sea más fácil detectar el origen de los posibles errores. Después de la integración de un módulo se llevan a cabo una serie de pruebas para probar que el funcionamiento es el correcto.

Existen dos técnicas básicas de hacer esta integración: (a) *descendente*, y (b) *ascendente*. En la técnica descendente los módulos se integran moviéndose hacia abajo por la jerarquía de control comenzando con el módulo principal, y siguiendo con los módulos subordinados en una estrategia *primero-en-profundidad* (siguiendo una secuencia de módulos encadenados hasta su final), o *primero-en-anchura* (no bajamos de nivel hasta que no se hayan integrado todos los módulos de un nivel determinado). Esto tiene la ventaja de que probamos las estructura de control principales primero, pero tiene el inconveniente de que tenemos que simular la actuación de los módulos de bajo nivel que no han sido integrados (los módulos simulados se denominan resguardos).

En la técnica de integración ascendente se empieza por la integración y prueba de los módulos de más bajo nivel agrupándolos generalmente en grupos que realizan una función específica. La ventaja de esta técnica es que elimina la necesidad de resguardos pero tiene el inconveniente de que el programa como entidad no existe hasta que se ha añadido el último módulo (Myers, 1979).

La mejor estrategia suele ser generalmente una estrategia de compromiso entre ambas técnicas, utilizando la aproximación descendente para los niveles superiores de la estructura del programa, y la ascendente para los niveles subordinados.

Durante la integración se pueden llevar a cabo pruebas de la caja blanca, pero las técnicas que más prevalecen son las de diseño de casos de *prueba de la caja negra*. Las pruebas de la caja negra se centran en la comprobación de que el producto cumple la función específica para la que fue diseñado. De esta forma se ignora la estructura de control y se concentra la atención en la detección de errores como: (1) funciones incorrectas o ausentes, (2) errores de interfaz, (3) errores en estructuras de datos o en accesos a bases de datos externas, (4) errores de rendimiento, y (5) errores de inicialización o terminación.

2.8.3. Prueba de validación

Durante las pruebas de unidad y las pruebas de integración realizamos tareas que comprenden básicamente a la fase de verificación del software, comprobando si se está construyendo el producto correctamente. También se pueden llevar a cabo tareas de validación pero generalmente estas se encuadran en lo que se conoce como *prueba de validación*.

En la prueba de validación comprobamos que el software funciona de acuerdo con las expectativas razonables del cliente. Estas expectativas se hallan contenidas en la especificación de requisitos del software, que contiene una sección denominada *Criterios de Validación*. La información contenida en esa sección forma la base de la aproximación a la prueba de validación.

La validación del software se consigue mediante una serie de pruebas de la caja negra en las que se comprueban que se satisfacen los requisitos funcionales, que se alcanzan los requisitos de rendimiento, que la documentación es correcta e inteligible y que se cumplen otro tipo de requisitos como portabilidad, compatibilidad, recuperación de errores, facilidad de mantenimiento, etc.

Es virtualmente imposible que el encargado de desarrollo del software pueda prever como un cliente usará realmente un programa. Por ello se llevan a cabo pruebas que requieren la participación del usuario final, como son las *pruebas alfa* y las *pruebas beta*. La prueba alfa es conducida por el usuario en el lugar de desarrollo. El programador se encuentra siempre presente registrando los distintos errores y problemas de uso que pueden aparecer. Las pruebas alfa se llevan a cabo en un entorno controlado. Las pruebas beta se llevan a cabo por los usuarios en sus lugares habituales de trabajo, es decir, entornos no controlados por el desarrollador del software que generalmente no está presente. El cliente registra todos los problemas que encuentra durante la prueba e informa regularmente al equipo de desarrollo. En base a estos informes se modifica el programa y se prepara la versión definitiva del software.

2.8.4. Prueba del sistema

El software no es algo aislado sino que forma parte de un sistema mayor de tratamiento de la información. Una vez que el software ha sido ampliamente validado es incorporado a otros elementos del sistema. La prueba del sistema cae fuera del alcance del proceso de ingeniería del software y no la realiza únicamente el desarrollador del software.

Dentro de las pruebas del sistema podemos destacar las siguientes:

1. *Pruebas de recuperación.* Fuerza el fallo del sistema de muchas formas y verifica que la recuperación se lleva a cabo apropiadamente. Si la recuperación es automática (llevada a cabo por el propio sistema) hay que evaluar sus mecanismos de inicialización, recuperación del estado, recuperación de los datos y rearranque. Si la recuperación requiere la intervención humana, hay que evaluar los tiempos medios de reparación para determinar si están dentro de unos límites aceptables.

2. *Pruebas de seguridad.* Verifica que los mecanismos de protección incorporados en el sistema lo protegen de la penetración impropia. El responsable de esta prueba ataca al sistema con todo tipo de medios, intentando demostrar que el coste de una entrada ilegal es mayor que el valor de la información obtenida.

3. *Pruebas de resistencia.* Se fuerza a los programas a situaciones anormales para comprobar su robustez. Consiste en ejecutar un programa de forma que demande recursos en cantidad, frecuencia o volúmenes anormales, y se evalúa el funcionamiento del sistema.

4. *Pruebas de rendimiento.* Se realiza en sistemas en tiempo real o sistemas empotrados en los que el software además de cumplir las funciones requeridas tiene que ajustarse a una serie de requisitos de rendimiento. Las pruebas de rendimiento van, a menudo, emparejadas con las pruebas de resistencia.

2.9. Resumen

En este capítulo hemos descrito los principales modelos de construcción del software convencional que existen en la ingeniería del software, así como las principales características que se incluyen en la validación de dichos sistemas.

Los primeros métodos de construcción del software consistían básicamente en ponerse a programar después de haber especificado brevemente el sistema (lo que se ha dado en llamar "codifica y corrige"). Ante los evidentes problemas que tenía esta técnica (código poco estructurado, difícil mantenimiento, etc.) surgió la idea de dividir el proceso de construcción del software en fases bien estructuradas. De esta forma nació el modelo de desarrollo más ampliamente conocido y utilizado: el modelo en cascada. Sin embargo, la rigidez de este modelo provoca que su aplicación sólo sea efectiva en dominios estables y bien conocidos.

Se han desarrollado varios tipos de métodos para paliar los problemas del modelo en cascada, la mayoría de ellos basados en un enfoque incremental, es decir, no hay porque desarrollar el sistema como un todo sino que se puede dividir el desarrollo en incrementos funcionales. Entre los modelos incrementales más populares está el modelo en espiral propuesto por Barry Boehm (1988).

En este capítulo también se realiza una introducción a la verificación y validación de sistemas convencionales que, generalmente, se dividen en varias fases como la prueba de unidad, la prueba de integración, la prueba de validación y la prueba del sistema.

Como veremos en los siguientes capítulos muchos de los modelos y métodos desarrollados para la ingeniería del software son válidos, con sus correspondientes modificaciones, en la ingeniería del conocimiento.

3. INGENIERÍA DEL CONOCIMIENTO

La imaginación es más importante que el conocimiento
Albert Einstein (Físico y matemático de origen alemán. 1.879 – 1.955)

Si la confusión es el primer paso hacia el conocimiento, yo debo de ser un genio
Larry Leissner

El conocimiento de ningún hombre puede ir más allá de su experiencia
John Locke (Filósosfo inglés. 1632 – 1704)

El verdadero conocimiento está en conocer que no conoces nada
Socrates (Filósofo griego. 470 – 399 a.c.)

Una vez que has acumulado suficientes conocimientos, eres demasiado viejo para recordarlos.
Anónimo

Mientras que los desarrolladores de software convencional han sido denominados con el término *ingenieros del software*, los desarrolladores de sistemas basados en conocimiento y sistemas expertos han pasado a denominarse *ingenieros del conocimiento*. La misión de la ingeniería del conocimiento es adquirir, formalizar, representar y usar grandes cantidades de conocimiento de alta calidad y específico de una tarea. (Borrajo et al., 1993). Este conocimiento se integra dentro de un sistema computacional para resolver problemas complejos que normalmente requieren un alto nivel de experiencia humana (Maté y Pazos, 1988).

Durante el desarrollo de un sistema experto, el ingeniero del conocimiento tiene que afrontar retos y problemas desconocidos para los ingenieros del software. Como su propio nombre indica, la ingeniería del conocimiento está fuertemente relacionada con el conocimiento implicado en la resolución de problemas. Por ello, los sistemas expertos exhiben ciertas características estructurales diferenciales que describiremos a continuación.

3.1. Estructura de un sistema experto

Un sistema experto es simultáneamente un elemento software y un modelo del conocimiento y del razonamiento humano que, como todos los modelos, nunca será perfecto. Esto añade una dificultad más a la hora de validar el sistema (O'Keefe y O'Leary, 1993).

Las diferencias en la estructura entre un sistema experto y un programa convencional se hacen más evidentes si vemos un diagrama de bloques de los principales elementos que forman un sistema experto, como se muestra en la Figura 3.1 (Geissman y Schultz, 1988).

En este diagrama distinguimos, en primer lugar, una *base de conocimientos* que contiene una descripción de cómo resolver determinados problemas, y hechos importantes que deben ser tenidos en cuenta. El conocimiento se expresa a través de un método de representación que puede variar mucho de un sistema a otro (el más común es la representación a través de reglas IF – THEN). La base de conocimientos es desarrollada por el ingeniero del conocimiento en base a la información obtenida por un experto humano del domino de aplicación.

El *motor de inferencias* es un programa encargado de dirigir el funcionamiento del sistema experto. Interpreta la información almacenada en la *memoria de trabajo* (que se encarga de contener una descripción del problema actual sobre el que se está trabajando) y selecciona los procedimientos a seguir en la base de conocimientos. Estos procedimientos cambiarán de nuevo la memoria de trabajo (creando nuevas hipótesis, rechazando otras, recogiendo nuevos datos, etc.) de forma que el motor de inferencias vuelve a ejecutar nuevos procedimientos. El proceso termina cuando hallamos la meta buscada o cuando ya no quedan más procedimientos que ejecutar. El motor de inferencias puede utilizar varios paradigmas de resolución de problemas como el encadenamiento progresivo, el encadenamiento regresivo, etc.

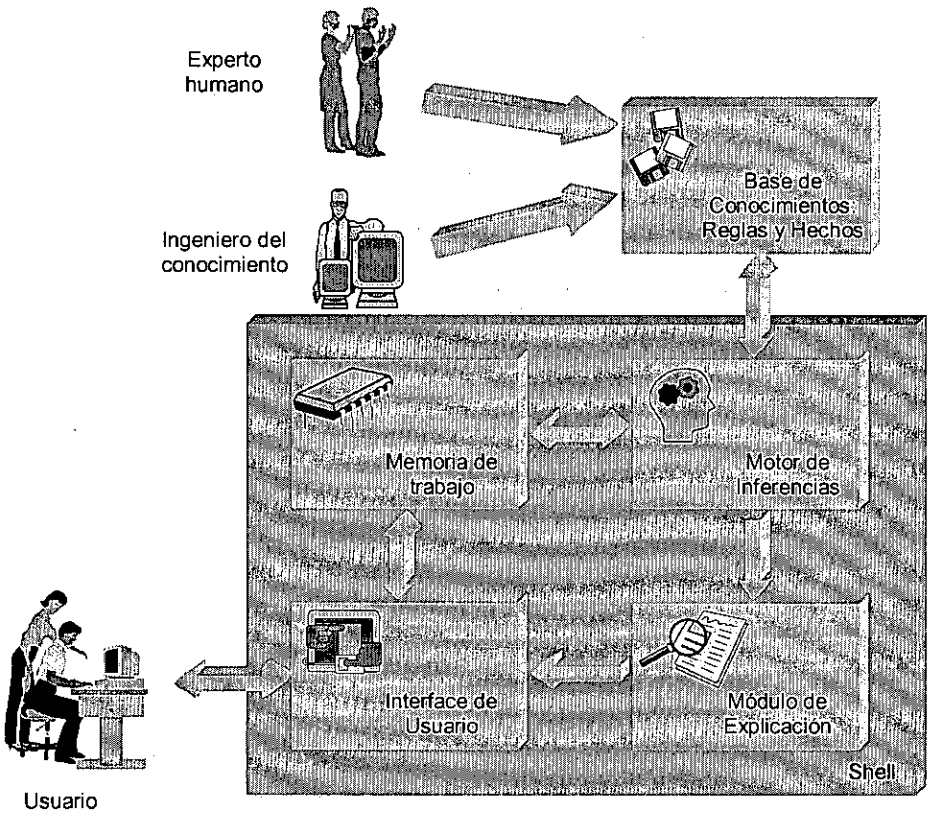


Figura 3.1. Diagrama de bloques de un sistema experto.

Por último tenemos el *interfaz de usuario,* que se encarga de conducir el diálogo con el usuario, requiriendo información y mostrando las conclusiones; y el *modulo de explicación,* que se utiliza para mostrar como se ha llegado a una determinada conclusión.

Todos los elementos comentados salvo, obviamente, la base de conocimientos pueden aparecer ya desarrollados en herramientas comerciales o "shells". Estas herramientas incorporan motores de inferencia configurables, facilidades de explicación, soporte de los esquemas de representación del conocimiento y, a menudo, utilidades para la construcción del interfaz. De esta forma la tarea del ingeniero de conocimiento puede centrarse exclusivamente en la captura del conocimiento experto que, por otro lado, es la tarea más complicada de la construcción de un sistema experto. El inconveniente de las shells comerciales es que le restan flexibilidad a la construcción de los sistemas expertos ya que tenemos que ceñirnos a las características de las mismas. Además estas herramientas pueden no ser adecuadas para sistemas en tiempo

real, y ser menos portables que los lenguajes procedimentales comunes, como el C, o lenguajes utilizados normalmente en la IA, como el LISP o el PROLOG. La analogía más directa de las shells de IA que podemos encontrar en el software convencional son los gestores de bases de datos, que facilitan el procesado y el almacenamiento estructurado de la información mientras que el usuario sólo tiene que ocuparse de introducir los datos que desee almacenar (Carrico et al., 1989).

Como vemos, una característica fundamental dentro de los sistemas expertos es la separación del conocimiento del domino de las estructuras de control que se encargan de manejar este conocimiento. Esto permite que el contenido de la base de conocimientos sea exclusivamente un modelo del conocimiento humano y que pueda ser comprensible por un experto humano del mismo domino. En el software convencional esto no tiene por qué ser así.

3.2. Problemas fundamentales de la ingeniería del conocimiento

Los principales problemas a los que se enfrenta un ingeniero del conocimiento a la hora de desarrollar un sistema experto son (Maté y Pazos, 1988):

a) *Adquisición del conocimiento*. Cómo trasladar el conocimiento humano, tal y como existe corrientemente en los textos y las mentes de los expertos humanos, a una representación abstracta efectiva.

b) *Representación del conocimiento*. Cómo representar el conocimiento en términos de estructuras de datos que una máquina pueda procesar.

c) *Mecanismos de razonamiento*. Cómo hacer uso de esas estructuras abstractas de datos para generar información útil en el contexto de un caso específico.

Veamos cada uno de estos puntos con un poco más de detalle.

3.2.1. Adquisición del conocimiento

La *adquisición del conocimiento* es una fase crucial en el desarrollo de los sistemas expertos que, sin embargo, no lo es tanto en los sistemas convencionales. Las únicas tareas de adquisición de conocimiento en un sistema convencional aparecen en forma de *determinación de requisitos* y son empleadas por el analista para delinear el problema que quiere resolver (McGraw y Harbison-Briggs, 1989). Por otro lado, en los sistemas expertos se entiende por adquisición del conocimiento (Figura 3.2) la construcción de un modelo computacional del comportamiento inteligente mediante la extracción del conocimiento de un experto (o una fuente de experiencia) y su transferencia a un programa o sistema experto (Buchanan et al., 1983).

La complejidad y la dificultad de la tarea de extracción del conocimiento ha provocado que muchos autores la definan como un "cuello de botella" que impide la amplia difusión de los sistemas expertos (Feigenbaum, 1979), (Platts, 1997).

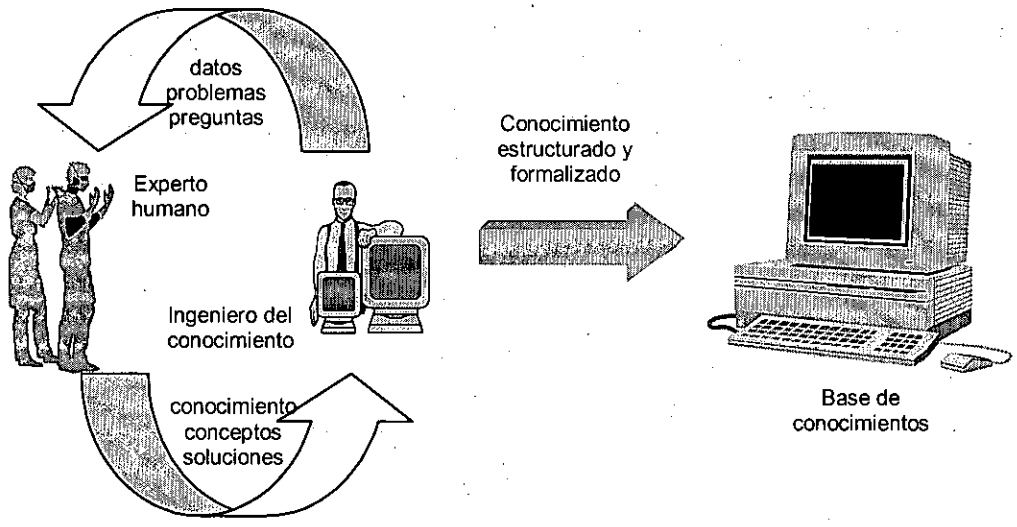


Figura 3.2. Proceso típico de adquisición del conocimiento.

La adquisición de conocimiento se suele dividir en dos fases (Gonzalez y Dankel, 1993): (1) *extracción* del conocimiento de las fuentes de experiencia y (2) *representación* de este conocimiento en una herramienta. Para la adquisición del conocimiento existen muchas técnicas (McGraw y Harbison-Briggs, 1989) que pueden resumirse en el esquema de la Figura 3.3. En ella presentamos cinco formas distintas de realizar la adquisición del conocimiento que varían desde técnicas manuales hasta técnicas con un alto grado de automatización.

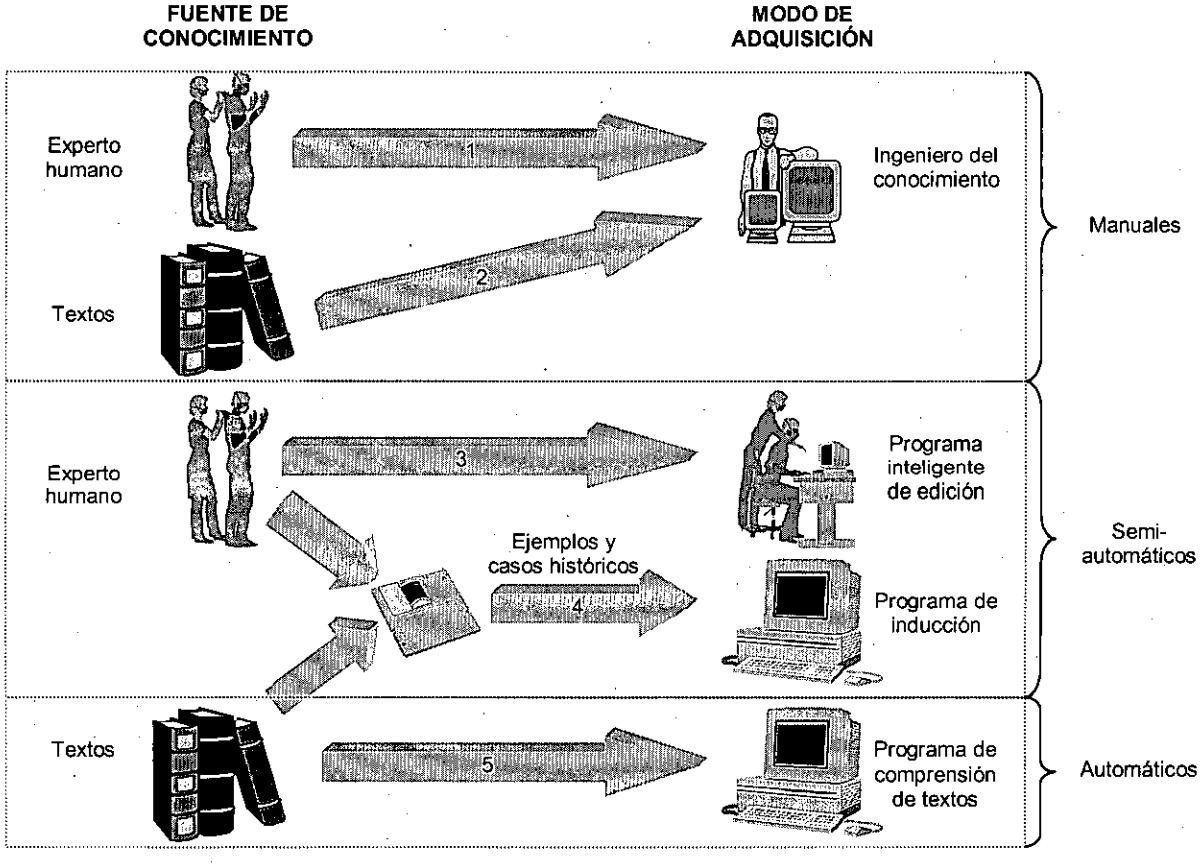


Figura 3.3. Posibles variaciones en los modos de adquisición del conocimiento.

En primer lugar tenemos la interacción directa entre el experto humano y el ingeniero del conocimiento. Esta interacción se llevará a cabo, generalmente, a partir de frecuentes entrevistas entre ambos. Como es normal, es muy difícil plantear las cuestiones adecuadas en la primera entrevista, y el proceso de adquisición se lleva a cabo a través de una serie de aproximaciones: el ingeniero codifica parte de lo que ha dicho el experto, se lo presenta para su modificación y surgen nuevas preguntas y respuestas que nos conducen a un nuevo modelo del sistema. El enfoque debe ser flexible y pragmático, porque ni el ingeniero del conocimiento ni el experto saben, al iniciar el desarrollo, cómo será al final el sistema de conocimiento. Los ingenieros de conocimiento ayudan a los expertos a descifrar cómo resuelven los problemas y a convencerles, a través de demostraciones de prototipos, que su conocimiento puede representarse de un modo útil (Harmon y King, 1985). Aquí es importante resaltar que muchas veces los expertos humanos saben resolver los problemas pero son incapaces de detallar de una forma clara el proceso que siguen para su resolución.

El proceso estructurado de extracción del conocimiento de los expertos humanos se denomina elicitación y generalmente viene acompañado del diseño y materialización de alguna estructura física (e.g., formularios, interfaces, etc.), con la que llevar a cabo dicha elicitación. Estas estructuras pueden ser posteriormente utilizadas en el proceso de validación (e.g., documentos de elicitación que se utilizan para adquirir casos comunes resueltos por el experto humano).

La entrevista no es el único modo de extracción de conocimiento de expertos humanos. Su combinación con otras técnicas como la observación directa o las técnicas intuitivas pueden hacer que esta extracción sea más efectiva (Gonzalez y Dankel, 1993). La *observación directa* consiste en la observación, por parte del ingeniero del conocimiento, del trabajo del experto mientras trata de comprender y duplicar sus métodos de resolución de problemas. En las *técnicas intuitivas*, el ingeniero del conocimiento trata de convertirse en un pseudo-experto intentando resolver los problemas del dominio con su pseudo-conocimiento y dejando que el experto humano critique su trabajo.

La adquisición del conocimiento no sólo se hace a partir de expertos humanos sino que normalmente suelen utilizarse otros tipos de fuentes de conocimiento como libros, artículos, manuales, videocasetes, etc. (opción 2 de la Figura 3.3).

En las dos primeras posibilidades de adquisición vistas hasta ahora el proceso es enteramente manual, y no existe ningún tipo de automatización. Sin embargo esta automatización ha ido introduciéndose paulatinamente en las tareas de adquisición como se puede ver en los modos 3 y 4 de la Figura 3.3. En el primero de éstos vemos cómo un experto interactúa con un programa inteligente de edición. De esta forma dicho experto puede introducir directamente su organización conceptual y sus heurísticas en la base de conocimientos. En este caso, el trabajo del ingeniero de conocimiento consiste en desarrollar convenientemente el programa inteligente de edición (que debe tener capacidades sofisticadas de diálogo y un amplio conocimiento de la estructura de la base de conocimientos). El ejemplo más destacado de esta técnica es el desarrollado por Davis (1976) en TEIRESIAS.

El cuarto modo de adquisición representado en la Figura 3.3 corresponde a una técnica conocida normalmente por aprendizaje automático o "machine learning" (Michie, 1982). Esta técnica consiste en la recolección de ejemplos o casos históricos a

partir de expertos humanos, o directamente de la bibliografía, y su introducción en un programa de inducción que nos permitirá extraer reglas y heurísticas de los casos. Esta técnica tiene la ventaja de que, aunque los expertos tengan dificultades en expresar su conocimiento, dichos expertos son particularmente receptivos a explicar sus técnicas a partir de ejemplos.

El último modo de adquisición presentado representa una idea futurista en la cual el proceso de adquisición es completamente automático. Así un programa de comprensión de textos "leería" los textos básicos del dominio, y a partir de ellos extraería las reglas necesarias para la construcción de la base de conocimientos. Esta técnica requiere una mayor sofisticación en los programas de comprensión del lenguaje natural que la existente hoy en día, así como capacidades en la comprensión de diagramas.

A pesar de los recientes avances en la automatización de los procesos de adquisición del conocimiento, la interacción con expertos humanos sigue siendo necesaria a la hora de desarrollar un sistema experto. Esto implica que el ingeniero del conocimiento no sólo tiene que tener conocimientos de programación clásica sino que además debe combinar grandes dosis de psicología cognoscitiva con técnicas de programación simbólica.

Por otro lado los programadores convencionales trabajan de una forma completamente distinta, en este caso no se intenta adquirir ningún conocimiento experto por lo que los programadores colaborarán más activamente con los usuarios del sistema. Generalmente esta colaboración se hará a nivel de diseño, una vez se ha logrado un diseño detallado la interacción entre los programadores y los usuarios no tiene por qué ser elevada. Aunque generalmente se recomienda que el diálogo con el usuario sea fluido, si el dominio es conocido y los objetivos están claros, dicho diálogo no es tan determinante para el éxito de la aplicación.

El hecho de que el conocimiento humano sea subjetivo por naturaleza puede complicar aún más las cosas. Por ejemplo, en algunas aplicaciones podemos hallarnos con el caso de que dos expertos de la misma categoría decidan resolver un determinado problema de dos formas diferentes. Las dos soluciones pueden ser correctas pero cada experto considerará la suya como la mejor y señalará la solución del otro experto como menos adecuada. Además los expertos humanos no tienen por qué ser siempre constantes en sus respuestas, factores externos como distracciones, cansancio, estrés, etc. pueden afectar a sus decisiones.

Esta subjetividad afecta también de forma importante a la validación del sistema, ya que el árbitro que decidirá finalmente el grado de corrección de un sistema experto será un experto humano (generalmente no es muy corriente la existencia de referencias estándar aceptadas por todos). Por ello es necesario llevar a cabo medidas que la corrijan estas desviaciones (validaciones ciegas, validaciones con múltiples expertos, etc.) que veremos más adelante. Otra complicación más que puede aparecer es que el experto puede interpretar la validación del sistema como una validación de su propio conocimiento (Shaw y Woodward, 1988). Sin embargo, en el software convencional la cuestión de la corrección de los resultados generalmente no es un problema, el encargado de la validación puede siempre determinar de forma clara si la respuesta ofrecida por el sistema es correcta o no.

Otro problema que también aparece es que, para construir un sistema experto, el ingeniero de conocimiento debe convertirse en un "casi-experto" del dominio de aplicación. En la programación convencional el desarrollador solo necesitaría las especificaciones del software para poder construir el código (Lethan y Jacobsen, 1987).

3.2.2. Representación del conocimiento

La elección del esquema de representación del conocimiento, así como de la herramienta que se va a utilizar para representarlo, es una tarea crítica dentro del desarrollo de los sistemas expertos que puede tener gran impacto en el posible éxito de dicho sistema.

Generalmente hay dos opciones no necesariamente excluyentes: (1) *esquemas procedimentales*, que abarcan a los sistemas que utilizan reglas de producción y a los sistemas basados en reglas lógicas y (2) *esquemas declarativos*, que abarcan a los sistemas que utilizan frames, objetos o redes semánticas, entre otros.

Los esquemas procedimentales representan el conocimiento en base a estructuras dinámicas que nos describen la forma en que se utiliza dicho conocimiento. Por otro lado, los esquemas declarativos son más adecuados para la representación de hechos estáticos interrelacionados entre sí, y con una limitada información sobre la forma de emplear dicho conocimiento.

Los sistemas que combinan las capacidades de representación de los métodos declarativos (generalmente frames y objetos) con las capacidades inferenciales de los métodos procedimentales (generalmente reglas de producción) suelen ser las mejores soluciones. Tienen el inconveniente de que son más complejos que un simple sistema basado en reglas y, por lo tanto, más difíciles de utilizar.

El esquema de representación elegido tendrá una influencia directa en la elección del mecanismo de razonamiento y de la herramienta de desarrollo. Sin embargo todas estas elecciones no son fáciles de realizar y puede aparecer el problema conocido como *desplazamiento del paradigma* o *paradigm shift* (Waterman, 1986).

El desplazamiento del paradigma puede aparecer durante la fase de desarrollo cuando el ingeniero del conocimiento descubre que el esquema de representación del conocimiento, la herramienta y/o otros aspectos del diseño, no son adecuados. Este descubrimiento es debido generalmente a una falta de comprensión de las complejidades del problema, o a una subestimación de su magnitud, y puede conllevar graves problemas en el desarrollo del sistema. El ingeniero del conocimiento se enfrenta entonces al dilema de continuar el desarrollo con infraestructuras inadecuadas que, posteriormente, pueden dar lugar a serias dificultades, o elegir el esquema de representación y/o otras características de forma adecuada, teniendo en cuenta el serio retraso que esto puede producir al proyecto. Si el desplazamiento del paradigma tiene lugar en etapas tempranas del desarrollo puede ser beneficioso porque ayuda a detectar errores y puede ser corregido sin un gran esfuerzo.

En la Figura 3.4 podemos ver una representación del desplazamiento del paradigma.

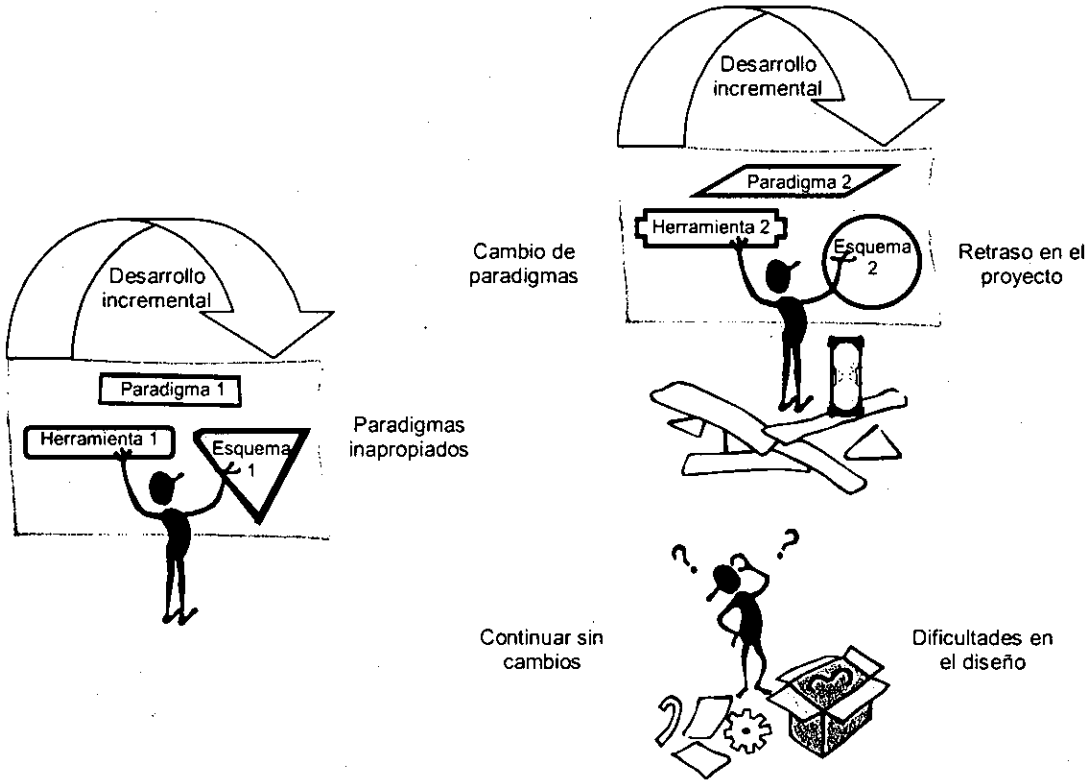


Figura 3.4. Desplazamiento del paradigma.

3.2.3. Mecanismos de razonamiento

Después de esbozar los mecanismos de adquisición y representación del conocimiento, es importante hacer mención a los mecanismos que permiten usar ese conocimiento adquirido y representado formalmente, para generar información útil en el dominio de aplicación en el que nos estamos moviendo.

Los mecanismos o modelos de razonamiento forman parte de las estructuras de control del conocimiento (que en los sistemas de producción son una parte del motor de inferencias) y son fundamentales a la hora de organizar la búsqueda de soluciones.

Las características del dominio, así como las características del problema a resolver condicionan, en gran medida, el modelo de razonamiento a escoger. Siguiendo esta idea presentamos una serie de dominios típicos y sus modos de razonamiento asociados (entendiendo que estas situaciones son ideales y siempre puede haber casos que presenten simultáneamente varias de las características presentadas).

a) *Dominios de naturaleza simbólica.* Se caracterizan porque en ellos puede encontrarse una solución de forma biunívoca. Los modelos de razonamiento empleados en este tipo de dominios son de naturaleza categórica.

b) *Dominios de naturaleza estadística.* No se puede encontrar una solución única al problema planteado y tendremos que averiguar cuál de las posibles soluciones encontradas es la más probable. Dentro de los modelos de razonamiento estadístico el mas utilizado es el esquema Bayesiano.

c) *Dominios con incertidumbre.* Existen dominios en los que aparecen incertidumbres inherentes a los datos del problema, a los hechos del dominio

o a los propios mecanismos inferenciales. Los modelos de razonamiento deben ser capaces de manejar correctamente esta incertidumbre. Dentro de estos modelos podemos destacar el modelo de factores de certidumbre de Shortliffe y Buchanan (Shortliffe et al., 1979), y la teoría evidencial de Dempster y Shafer (Dempster, 1967), (Shafer, 1976).

d) *Dominios que incluyen matices de carácter lingüístico.* Los problemas del mundo real suelen expresarse a través de símbolos y no de números. La conversión entre un formato simbólico y un formato numérico no es una tarea sencilla. Entre los modelos de razonamiento que se encargan de manejar correctamente los matices lingüísticos destacan los basados en los conjuntos difusos.

El concepto de la incertidumbre juega un papel muy importante en los sistemas expertos. Esto es debido a que muchas tareas que requieren una conducta inteligente tienen asociado un cierto grado de incertidumbre. Los sistemas basados en conocimiento presentan una conducta inteligente, modelando las asociaciones empíricas y las relaciones heurísticas que los expertos han ido formado, a partir de la práctica diaria en lugar de utilizar algoritmos determinísticos que ofrezcan soluciones precisas. En consecuencia los sistemas basados en conocimiento deben ser capaces de trabajar con incertidumbre.

Los tipos de incertidumbre que pueden aparecer en sistemas basados en conocimiento guardan relación con los datos de los que se dispone, como por ejemplo:

a) Hay falta de información (algunos datos no están disponibles).

b) Los datos están presentes pero son ambiguos debidos a errores en las medidas, a la presencia de múltiples medidas contradictorias, etc.

c) La representación de los datos es imprecisa o inconsistente.

d) Los datos están basados en suposiciones del usuario.

e) Los datos están basados en estándares pero estos estándares presentan excepciones.

Por otra parte, la incertidumbre podría venir del propio conocimiento heurístico ya que:

a) Representa suposiciones que los expertos hacen basándose en asociaciones plausibles que han observado.

b) Puede no ser aplicable en todas las situaciones que se presenten.

Debido a todas estas posibles causas de error, la mayoría de los sistemas basados en conocimiento requieren la incorporación de alguna forma de tratamiento de la incertidumbre. Al trabajar con incertidumbre se deben tener en cuenta los siguientes aspectos:

a) El modo de representar el conocimiento incierto.

b) La forma de combinar datos inciertos.

c) La forma de inferir conocimiento a partir de datos inciertos.

Todas estas cuestiones son de gran importancia dado que pueden tener una gran influencia en el modo de operar del sistema y en las conclusiones finales obtenidas. Numerosos investigadores sostienen que la limitada capacidad de los sistemas basados en conocimiento, respecto al tratamiento de la incertidumbre, restringe en gran medida su rendimiento y es por ello que se están realizando grandes esfuerzos en la búsqueda de modelos más avanzados de manejo de conocimiento impreciso (Gonzalez y Dankel, 1993).

3.3. Diferencias entre la ingeniería del conocimiento y la ingeniería del software

Aparte de las diferencias ya mencionadas, al hablar de la estructura de los sistemas expertos, y al describir los principales problemas que afronta la ingeniería del conocimiento, existen una serie de diferencias entre los programas convencionales y los sistemas expertos que es importante destacar.

En cuanto a los tipos de problemas apropiados para la resolución por programas convencionales o por sistemas expertos tenemos que, por un lado, el software clásico resuelve problemas que están bien definidos, que pueden ser especificados sin ambigüedad, y que son resueltos por algoritmos específicos. Por otro lado los sistemas expertos resuelven problemas que no están bien definidos, no pueden ser especificados con precisión, y son resueltos utilizando conocimiento heurístico (López et al., 90).

Los dominios en los que se va a instalar un sistema experto suelen ser dominios en los cuales nunca se ha intentado aplicar una solución computacional, por lo que no existe una experiencia previa de cómo tratar el problema y puede haber un ambiente hostil hacia la instalación de un sistema experto.

En cuanto a la forma de resolver los problemas, el software clásico utiliza métodos procedimentales y determinísticos para la resolución de problemas, mientras que los sistemas expertos utilizan métodos declarativos y no determinísticos que hacen un amplio uso de las heurísticas. El hecho de no utilizar algoritmos clásicos es debido a que, generalmente, en caso de existir serían tan costosos computacionalmente que no serían viables. Estas diferencias provocan que los sistemas expertos posean arquitecturas no secuenciales en las que los constantes efectos laterales y reacciones en cadena dificultan el seguimiento de la secuencia de ejecución. Esto no sucede así en los sistemas convencionales en los que resulta más fácil trazar la línea de ejecución del programa.

Además, los sistemas expertos pretenden resolver los problemas de la misma forma en que lo haría un experto humano en el mismo domino, es decir, pretenden modelar el conocimiento de un experto humano. Así un sistema que llegue a conclusiones correctas a través de líneas de razonamiento incorrectas puede no ser deseable. Este problema no aparece en los sistemas convencionales en los que se pretende capturar métodos estructurados y rutinarios. En palabras de Carrico et al. (1989) "los sistemas convencionales se centran en la manipulación de los datos mientras

que los sistemas expertos se centran en su interpretación y en el descubrimiento de relaciones simbólicas emulando el pensamiento humano".

De esta forma podemos ver que, generalmente, los sistemas convencionales se basan en la resolución de problemas a través del manejo de información almacenada en bases de datos y mediante procesos predecibles, fiables y exactos, que se ejecutan de forma rápida. Los sistemas expertos se basan en la evaluación de situaciones teniendo en cuenta aspectos como la abstracción o la incertidumbre. Generalmente estos sistemas son adaptables a distintos tipos de situaciones, y presentan capacidades de aprendizaje y capacidades de reconocimiento de patrones complejos.

Otra diferencia en la forma de resolver los problemas es que los sistemas expertos suelen ser altamente interactivos con el usuario, y suelen incluir rutinas de explicación de las conclusiones alcanzadas. En los sistemas convencionales no siempre es necesaria esta interactividad y no es común la inclusión de rutinas de explicación (Harmon y King, 1985).

Pero no sólo son diferentes las fuentes de conocimiento en los sistemas expertos y en los sistemas convencionales, sino que además la información utilizada por ambos es de distinto tipo. Así los sistemas expertos utilizan información numérica y simbólica, mientras que los sistemas convencionales utilizan preferiblemente información numérica.

Esto afecta a los procesos de verificación y validación ya que las técnicas empleadas usualmente para la validación de información numérica no son aplicables a los sistemas expertos, y se hace necesaria la inclusión de nuevas técnicas que permitan la validación de la información simbólica. Por ejemplo un modelo puramente cuantitativo producirá un valor numérico que puede ser contrastado con una determinada observación (la diferencia entre los dos valores nos dará una estimación de la exactitud del modelo). En contraste con esto tenemos, por ejemplo, una página de texto en la que describen prescripciones terapéuticas. La comparación de este texto con otro tomado como referencia requiere mucha experiencia, y la medición de las diferencias entre ambos es una tarea difícil (O'Keefe y O'Leary ,1993). Kulikowski y Weiss (1982) discutieron este problema en el contexto del sistema de diagnosis médica CASNET.

Además, como sabemos, los sistemas expertos se caracterizan porque sus soluciones no tienen por qué ser exactas, y suelen aceptar ciertos grados de incertidumbre. En los sistemas convencionales no se permite esta incertidumbre y menos en sistemas de análisis numéricos o en simulaciones. La incertidumbre añade un problema más a la validación ya que el número de estados a tener en cuenta aumenta considerablemente.

Un resumen de las diferencias entre sistemas expertos y software convencional puede verse la Tabla 3.1.

	Sistemas Expertos	Software convencional
Estructura	Separación del conocimiento de las estructuras de control	Separación de datos y algoritmos que utilizan los datos
	Suelen incluir estructuras de explicación de las conclusiones	No existen estructuras de explicación
	Se suelen construir a partir de herramientas ("shells") comerciales que permiten centrarse en el conocimiento	Existen gestores de bases de datos que nos permiten centrarnos exclusivamente en los datos y no en su almacenamiento o estructuración
Problemas apropiados	Problemas mal definidos, que no pueden ser especificados con precisión y que son resueltos utilizando conocimiento heurístico.	Problemas bien definidos, que pueden ser especificados sin ambigüedad y que son resueltos por algoritmos específicos.
	Generalmente dominios sin experiencia computacional	Generalmente dominios con experiencia computacional
Estrategias de resolución	Métodos declarativos y no deterministicos	Métodos procedimentales y deterministicos
	Intentan seguir líneas de razonamiento similares a las de los expertos humanos	Se centran en la solución y no en la forma en que se obtiene.
	Interpretan datos	Manipulan datos
	Tienen en cuenta aspectos como la abstracción, la incertidumbre, el aprendizaje, etc.	Resuelven problemas a través del manejo de información almacenada en bases de datos y mediante procesos predecibles, fiables y exactos.
	Son altamente interactivos	No siempre es necesaria la interactividad
Naturaleza del conocimiento empleado	Conocimiento proveniente de la experiencia humana (alta interacción con expertos)	Conocimiento de naturaleza algorítmica (alta interacción con usuarios)
Tipo de información utilizada	Información numérica y simbólica	Información numérica
	Información con incertidumbre	Información sin incertidumbre

Tabla 3.1. Diferencias entre los sistemas expertos y el software convencional.

3.4. Metodología de construcción de un sistema experto

Una de las principales carencias que acusaba la ingeniería del conocimiento en sus orígenes era la ausencia de una metodología de desarrollo comúnmente aceptada. Los sistemas expertos también son software, por lo que es fácil pensar que para su construcción y su validación se pueden seguir los mismos métodos vistos para los sistemas convencionales. Sin embargo las diferencias existentes entre el software convencional y los sistemas expertos provocan que las metodologías clásicas no sean del todo apropiadas para los métodos de la ingeniería del conocimiento. En palabras de Morris (1985) "El mundo de la inteligencia artificial no encaja bien en los entornos grises de la ingeniería del software".

La metodología en cascada, la más popular dentro de la ingeniería del software, representa un esquema de desarrollo lineal en el cual las iteraciones son debidas a fallos no previstos y representan cambios costosos en el sistema (Macleish, 1986). Sin embargo los sistemas expertos presentan características que impiden este desarrollo lineal. Geissman y Schultz (1988) declaran que el particular domino de aplicación de los sistemas expertos provoca que generalmente éstos empiecen con objetivos vagos, y sea muy difícil establecer los requisitos del sistema. Además de esta dificultad se plantea el hecho de que los sistemas expertos evolucionan a lo largo del tiempo y suelen aplicarse en entornos cambiantes. Todas estas características provocan que no sea posible aplicar técnicas clásicas de desarrollo, sino aplicar una metodología que consiste en un desarrollo incremental a través de prototipos (O'Keefe y O'Leary, 1993). A estas dificultades se suele añadir la falta de experiencia que los programadores suelen tener

cuando trabajan con sistemas expertos. Esto suele provocar que intenten solucionar problemas con técnicas o herramientas más propias de los sistemas convencionales (Noblett y Jones, 1991).

Veamos ahora en detalle las distintas metodologías que se han empleado para el desarrollo de sistemas expertos.

3.4.1. Método "adquiere y codifica"

Los primeros sistemas expertos fueron desarrollados sin un esquema preciso y bajo un método que pasó a llamarse "adquiere y codifica", muy similar al "codifica y corrige" del software convencional. Este método consiste en desarrollar el sistema en base a una serie de iteraciones, en cada una de las cuales se interactúa con el experto y se codifica el conocimiento extraído (McGraw y Harbison-Briggs, 1989).

En un principio se puede suponer que este sistema es válido, ya que las diferencias existentes con los sistemas convencionales impiden la utilización de métodos de desarrollo típicos. Esto puede ser cierto para sistemas pequeños y que actúan en solitario. Sin embargo, como su documentación es mínima, y se han sustituido las especificaciones y el diseño por el código, este método no es apto para sistemas de tamaño mayor y que formen parte de un proyecto más extenso.

Además, con el método "adquiere y codifica", estamos dejando de lado lecciones que los desarrolladores de software han estado aprendiendo durante 40 años. Por ejemplo, durante la fase de análisis los desarrolladores determinan el ámbito del problema, investigan visiones alternativas de las tareas a realizar y producen una especificación de requisitos del software con la que todo el personal puede trabajar. Viendo el proyecto como un todo, el personal debe realizar planificaciones y ayudas a la comunicación. Finalmente, para asegurarnos de que se han cumplido los requisitos, el personal debe establecer una metodología que permita la producción, validación y verificación de los productos intermedios (McGraw y Harbison-Briggs, 1989).

Boehm (1983) realizó una lista de principios que deberían cumplirse a la hora de desarrollar sistemas convencionales y que son aplicables a los sistemas expertos:

a) Desarrollar el sistema mediante un ciclo de vida dividido en fases.

b) Verificar y validar los resultados de cada fase.

c) Mantener el control del producto a través de "hitos" o puntos de control (milestones).

d) Utilizar técnicas modernas de programación como las herramientas y los análisis estructurados (por ejemplo, las herramientas CASE).

e) Mantener una descripción detallada de la situación del proyecto en cada momento.

f) Utilizar menos gente pero con más experiencia.

g) Comprometerse a mejorar el proceso adoptando diferentes métodos y técnicas.

Martin (1987) indicó la idoneidad de estos puntos en el desarrollo de sistemas de inteligencia artificial y también mostró como la técnica de "adquiere y codifica" sólo cumplía dos de ellos: la validación continua y la utilización de equipos pequeños.

3.4.2. Método de Buchanan.

Uno de los primeros métodos de desarrollo estructurado de sistemas expertos fue el desarrollado por Buchanan y otros autores en (Buchanan et al., 1983). Según estos autores la adquisición del conocimiento de un sistema experto (y por extensión todo el sistema) podía dividirse en cinco fases (Figura 3.5): identificación, conceptualización, formalización, implementación y prueba.

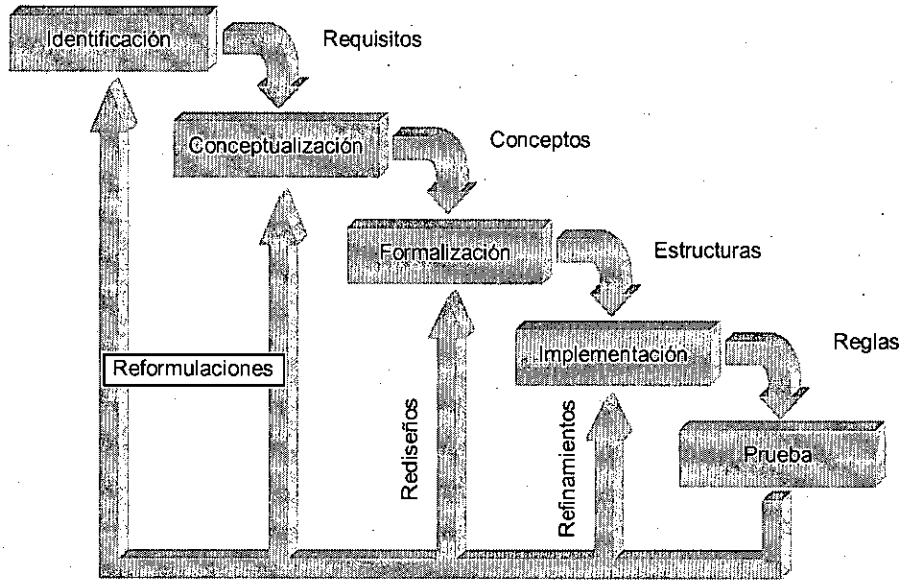


Figura 3.5. Modelo de Buchanan.

Sin embargo el proceso no está tan bien definido como puede sugerir la figura, y más bien representa una aproximación a las distintas y complejas fases que se llevan a cabo a la hora de desarrollar un sistema experto, y que pueden variar de una situación a otra.

La descripción de cada una de estas fases sería la siguiente:

- *Identificación.* Determina aspectos importantes del problema como los participantes (expertos del dominio, ingenieros del conocimiento y futuros usuarios), las características del problema (tipo, subtareas de que se compone, terminología a utilizar, aspectos fundamentales, etc.), los recursos disponibles (fuentes de conocimiento, facilidades computacionales, tiempo de desarrollo, financiación, etc.) y las metas a alcanzar (formalizar conocimiento experto, distribuir experiencia, ayudar a la formación de nuevos expertos, etc.).
- *Conceptualización.* Trata de poner el conocimiento dentro de un esquema conceptual. El experto y el ingeniero del conocimiento tratan de encontrar

conceptos que representen el conocimiento del experto, y determinar cómo es el flujo de información durante el proceso de resolución de problemas.

- *Formalización*. Esta fase consiste en el mapeado de los conceptos clave, de los subproblemas y de las características del flujo de información identificadas durante la fase anterior, en representaciones formales basadas en herramientas o esquemas de la ingeniería del conocimiento.
- *Elicitación*. Aunque no aparece en el trabajo original de Buchanan, es común incluir una fase de elicitación después de la fase de formalización. En esta fase se lleva a cabo la extracción del conocimiento mediante un soporte físico que es consistente con la información obtenida durante los procesos de identificación y conceptualización.
- *Implementación*. El ingeniero de conocimiento formula reglas y estructuras de control que representan los conceptos y el conocimiento formalizado. El resultado es un programa prototipo que nos permite comprobar si hemos conceptualizado y formalizado bien el conocimiento que el experto tiene sobre el problema.
- *Prueba*. Consiste en la evaluación del rendimiento del prototipo para encontrar errores o anomalías en la base de conocimientos y/o en las estructuras de inferencia.

Buchanan et al. posicionan los lazos de realimentación después de la fase de prueba pero también indican que el proceso no tiene porque seguir estrictamente la secuencia representada en la Figura 2.8. Autores posteriores como Mayrhauser (1990) señalan que las retroalimentaciones pueden aparecer entre cualquier par de fases de la metodología. Así, por ejemplo, si el ingeniero del conocimiento no encuentra reglas adecuadas durante la implementación puede requerir una vuelta atrás y una reformulación del problema. La nueva representación del ciclo de vida de los sistemas expertos sería tal y como se presenta en la Figura 3.6, una red completamente comunicada. Este tipo de estructura es muy compleja de controlar y de manejar, ya que el número de iteraciones entre las fases es desconocido, y los objetivos pueden cambiar a medida que avanza el desarrollo. También es difícil llevar a cabo un control de los progresos realizados.

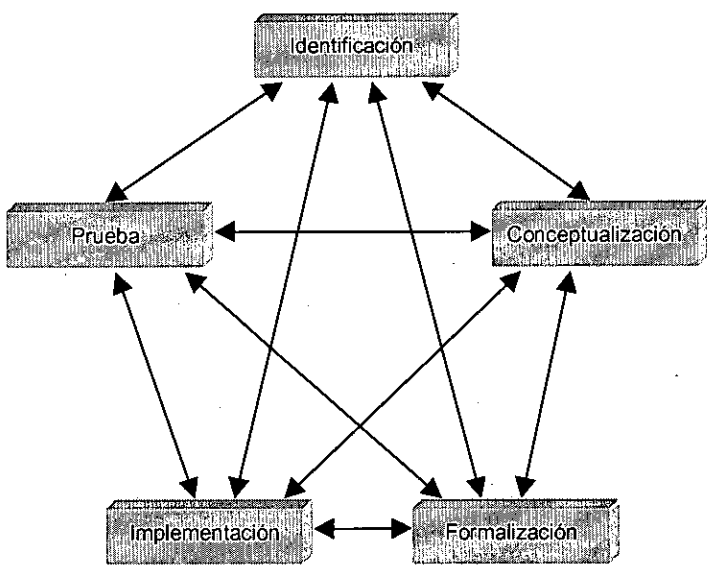


Figura 3.6. Ciclo de vida de los sistemas expertos según Mayrhauser (1990).

3.4.3. Diseño incremental

Como vemos, la construcción de sistemas expertos pone un énfasis especial en el desarrollo iterativo de los mismos, es decir, el sistema se desarrolla en base a una serie de ciclos en cada uno de los cuales se lleva a cabo un *refinamiento* (depurando errores en la base de conocimientos haciéndola más exacta), o una *extensión* del sistema existente (ampliando las capacidades del mismo) (Scott et al., 1991).

Esta tendencia hacia diseños incrementales o evolutivos es propia también de la ingeniería del software convencional (Noblett y Jones, 1991). Así en el apartado 2.1 de este trabajo veíamos cómo los más modernos desarrollos del software tendían a técnicas incrementales o evolutivas, dejando cada vez más de lado el clásico modelo en cascada. En palabras de Boehm (1989) "el modelo en cascada no está muerto, pero debería estarlo".

Los modelos de desarrollo incremental de los sistemas expertos se caracterizan porque intentan ajustar la terminología de la ingeniería del software al desarrollo de sistemas de inteligencia artificial. En el apartado anterior, cuando describíamos la metodología de Buchanan, veíamos que aparecían términos completamente nuevos (conceptualización, formalización, etc.) y que generalmente no habíamos visto en los modelos de desarrollo del software convencional. Esto es así porque Buchanan inicialmente pensó su metodología como una metodología de adquisición del conocimiento, e hizo hincapié en las fases que describen el proceso que sufre la información, desde que fluye del experto, hasta que es finalmente implementada en el sistema. Posteriormente este proceso se tomó como la construcción del sistema experto completo, porque las fases descritas se ajustaban también a esta descripción. En este caso las etapas de adquisición del conocimiento, propiamente dichas, serían las de conceptualización, formalización y elicitación.

Posteriormente, y probablemente con el fin de hacerlos más familiares, los métodos de desarrollo adquirieron los nombres típicos de la ingeniería del software (análisis, especificación, diseño, etc.) quedando las fases propias de los sistemas expertos, como la adquisición de conocimiento, imbuidas dentro de estas fases típicas. Existen multitud de métodos de desarrollo de sistemas expertos (probablemente tantos como investigadores trabajan en el tema) y la gran mayoría se basan en el prototipado rápido y el desarrollo incremental como paradigmas para lograr un sistema efectivo. Como ejemplos de los métodos de desarrollo incremental presentamos los trabajos de Gonzalez y Dankel (1993) y de Scott et al. (1991).

3.4.3.1. Método de Gonzalez-Dankel

El método de Gonzalez-Dankel (1993) se basa en una primera adquisición y representación del conocimiento, necesaria para la implementación de un aspecto limitado del dominio del problema. De esta forma se puede construir un primer prototipo que exhiba cierta semblanza con lo que será el sistema final. La construcción de este prototipo permite la aparición de información de retroalimentación que nos ayuda a definir el ámbito de nuestro conocimiento, las necesidades del usuario y la validez de las decisiones tomadas durante la etapa de diseño. De esta forma, si fuese necesario realizar un desplazamiento del paradigma, su impacto sería mínimo debido a su temprana aparición.

Este prototipo inicial utiliza un ciclo de vida modificado. Las fases de análisis y especificación se realizan teniendo en cuenta el sistema global, pero el diseño y la implementación se realizan de forma más sencilla y preliminar. De esta forma podemos obtener pronto un prototipo que puede ser evaluado para obtener la necesaria información de retroalimentación. El prototipo inicial puede entonces desecharse, o ser mejorado de forma incremental, hasta desarrollar un subsistema del producto final. Muchas veces se prefiere desecharlo para empezar el proceso de desarrollo de una forma fresca, evitando los errores iniciales que hayamos podido producir.

El proceso que sigue es el del desarrollo incremental del sistema. Este desarrollo se centra en el concepto de "divide y vencerás", en donde el conocimiento se separa en módulos que son desarrollados de forma incremental hasta componer el problema completo. Este desarrollo implica varios ciclos de elicitación de conocimiento de los expertos, implementación de este conocimiento en los expertos, validación de los resultados y refinamiento de la implementación o corrección de los errores encontrados.

Al utilizar la división en módulos podemos tener al sistema parcialmente funcionando antes de estar completamente terminado. Generalmente esto no es posible hacerlo en los sistemas convencionales debiendo esperar a que estén totalmente implementados antes de empezar a usarlos.

El método de Gonzalez y Dankel se representa en el esquema de la Figura 3.7.

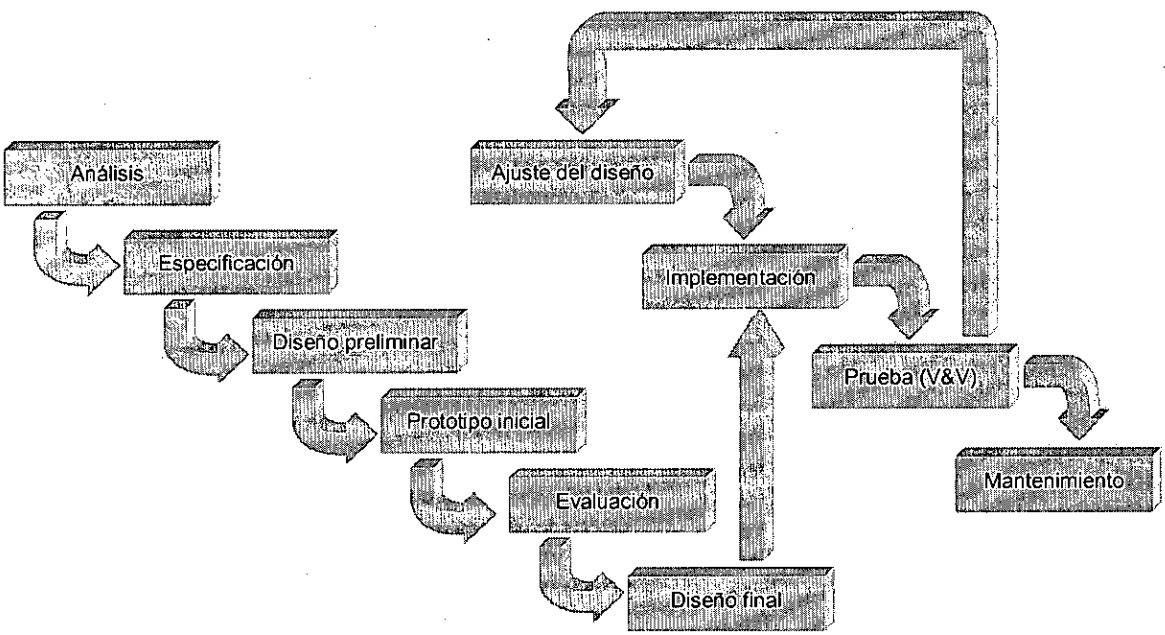


Figura 3.7. Método de desarrollo de sistemas expertos de Gonzalez y Dankel (1993).

Esta descripción coincide con un modelo de desarrollo del software convencional que incluya el prototipado rápido y el desarrollo incremental. Sin embargo las fases descritas incorporan ciertas diferencias de forma que admitan las características diferentes de los sistemas expertos, como veremos a continuación:

1. *Análisis del problema.* Evalúa el problema y los recursos disponibles para determinar la aplicabilidad de la solución basada en el conocimiento. Se realizan análisis de coste-beneficios, y también pueden establecerse estudios de mercado.

2. *Especificación de requisitos*. Se formaliza y se escribe lo aprendido durante la fase de análisis. Se fijan los objetivos del proyecto y los medios a utilizar para conseguir esos objetivos. Debido a las especiales características de los sistemas expertos suele ser complicado establecer unos requisitos claros desde el primer momento; sin embargo, la experiencia indica que los sistemas no pueden desarrollarse adecuadamente sin estar basados en unas especificaciones formales.

3. *Diseño preliminar*. Trata las decisiones de alto nivel necesarias para el desarrollo del prototipo inicial. Se determina el esquema de representación del conocimiento, la elección de la herramienta y la elección de los expertos humanos que colaborarán en el desarrollo. En esta fase es necesaria la realización de tareas de adquisición del conocimiento.

4. *Prototipo inicial y evaluación*. Se desarrolla un prototipo similar al sistema completo pero con una funcionalidad limitada. Este prototipo suele incluir una interfaz desarrollada y un subconjunto de conocimiento razonablemente robusto. Mediante el prototipo se pretende extraer nuevo conocimiento de los expertos y validar las decisiones de diseño establecidas. Se realiza una evaluación para comprobar que no se han cometido errores en el diseño preliminar.

5. *Diseño final*. Implica la selección de las herramientas y los recursos necesarios para el desarrollo del sistema (también incluye la selección del esquema de representación del conocimiento). Se deben especificar cuáles son los módulos en los que se va a dividir el sistema, cuáles son sus entradas y cuáles son las salidas que se pretenden obtener.

6. *Implementación*. Sigue las indicaciones obtenidas del diseño para implementar la base de conocimientos del sistema.

7. *Prueba*. Generalmente conocido en el mundo de los sistemas expertos como la fase V&V (verificación y validación). Los objetivos son similares a los de la fase de test en los sistemas convencionales pero la forma de llevarla a cabo difiere considerablemente. Esto se verá con más detalle en apartados posteriores de este trabajo.

8. *Ajuste del diseño*. A medida que el trabajo progresa es necesario realizar ajustes al diseño al principio de cada iteración. Si estos ajustes son menores no hay problema, pero si los ajustes requieren cambios significativos pueden aparecer desplazamientos del paradigma.

9. *Mantenimiento*. Fase similar a la descrita para los sistemas convencionales.

3.4.3.2. Método de Scott.

En este método, el desarrollo de un sistema experto se divide en cuatro fases (Scott et al., 1991): (a) *fase de análisis*, en la que las partes interesadas investigan la posibilidad de desarrollar un sistema experto, (b) *fase de especificación*, en la que se inicia el proyecto y se fijan las bases a utilizar en el desarrollo, (c) *fase de desarrollo*, en la que se realiza el diseño y la implementación del sistema y (d) *fase de utilización*, en

la que se habilita el sistema para su uso rutinario. Estas fases se dividen en subfases como puede verse en la Figura 3.8.

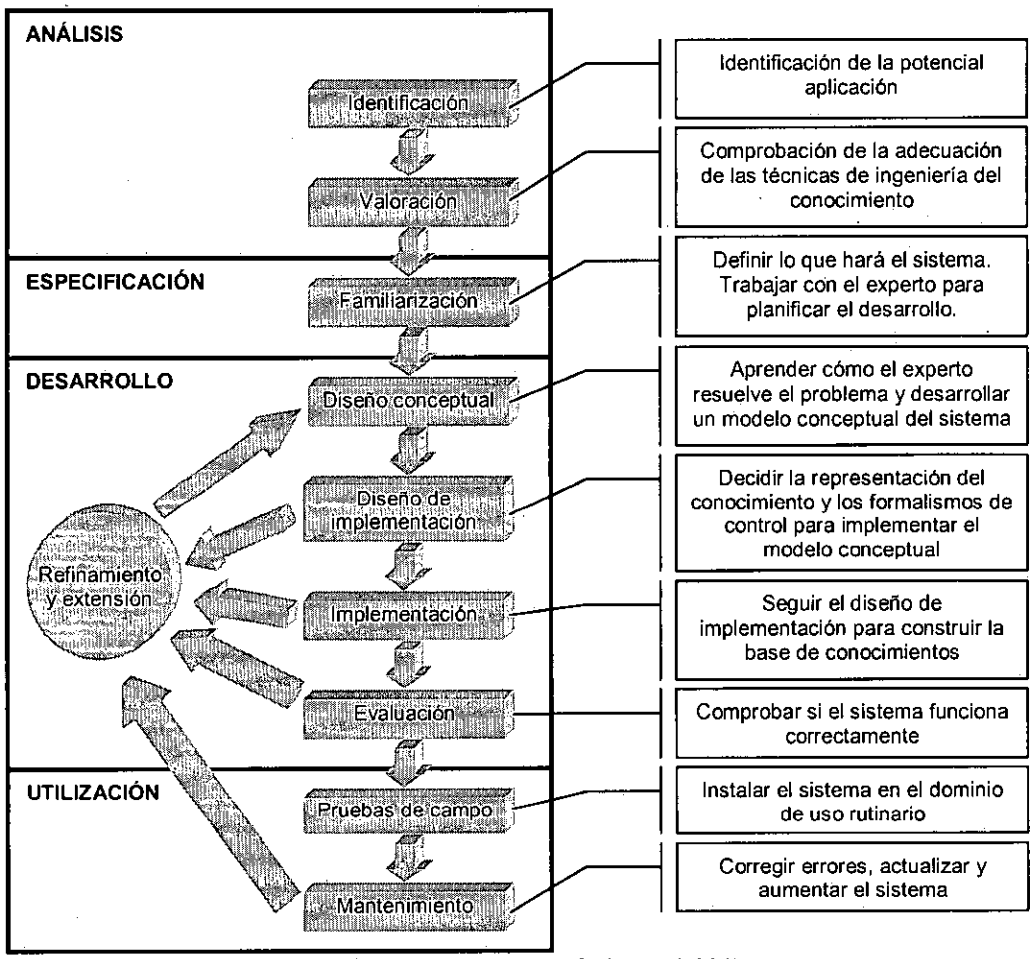


Figura 3.8. Método de desarrollo de sistemas expertos de Scott. (1991).

Los aspectos importantes de esta metodología son los siguientes:

- Sigue haciendo hincapié en el prototipado rápido y en el desarrollo incremental. Las primeras versiones del sistema no tienen por qué realizar todas las tareas posibles, ni por qué tratar todos los conjuntos de casos. Los incrementos posteriores se realizarán a través de una fase de refinamiento y extensión.
- Los sucesivos prototipos que se van formando son una ayuda para el proceso de adquisición del conocimiento.
- La fase de utilización empieza cuando el sistema se instala en el dominio en el que se usará de forma rutinaria. La fase de mantenimiento posterior puede mostrar errores o sugerencias de los usuarios, que es necesario corregir e implementar (pueden modificarnos el diseño del sistema si los cambios a realizar son muy grandes).

Como vemos las características son muy parecidas a las de la metodología de Gonzalez y Dankel, sólo que la forma de representar las fases es diferente. Sin embargo Scott et al. prestan más atención a la fase de adquisición del conocimiento. Aunque esta fase no aparece en la Figura 2.11 es un proceso que se distribuye en todas las fases que se han representado, es decir, la adquisición del conocimiento tiene cierta importancia

en cada una de las fases de desarrollo de un sistema experto. Esta importancia viene determinada por la fase en la que nos encontremos según se muestra en la Figura 2.12.

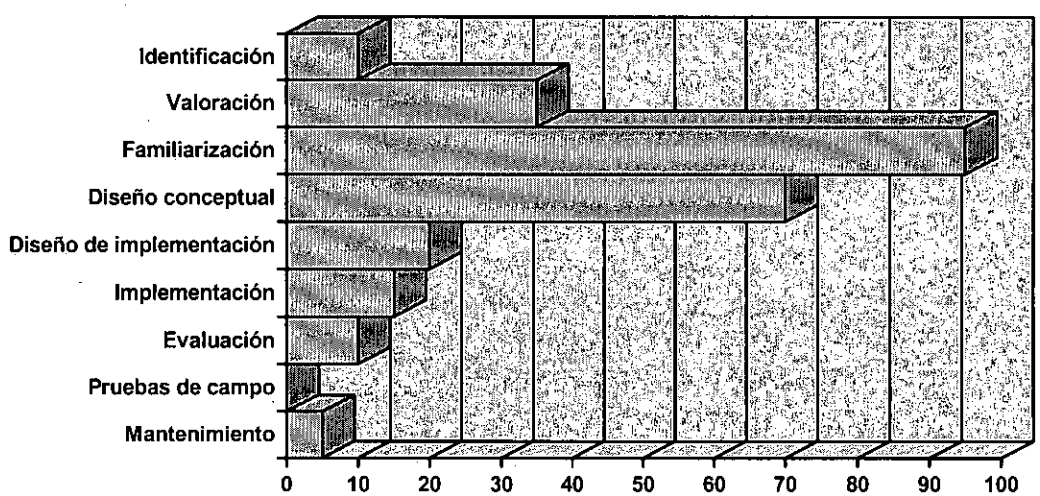


Figura 3.9. Porcentaje del tiempo total que es empleado en adquisición del conocimiento en cada fase.

Scott et al. diferencian dos tipos de adquisición del conocimiento:

1. *Adquisición inicial.* Es una fase preparatoria en la que la información obtenida nos permite tener un conocimiento más amplio de lo que debe hacer el sistema experto, de cómo va a ser usado, y de cómo hay que desarrollarlo. Esta adquisición inicial aparece en las fases de análisis y especificación.

2. *Adquisición detallada.* Se caracteriza porque su foco es más estrecho y profundo, es decir, pone más énfasis en los detalles que la fase anterior. La información obtenida en esta fase permite, a los ingenieros del conocimiento, comprender como los expertos humanos realizan sus tareas. Esta comprensión permite que se trasladen los procedimientos de los expertos humanos a la base de conocimientos de un sistema experto. La adquisición detallada aparece en las fases de desarrollo y utilización.

3.4.3.3. Tipos de prototipos

Como hemos visto en las metodologías expuestas en los apartados anteriores, el desarrollo incremental consiste en la construcción de una serie de prototipos que son modificados sucesivamente hasta obtener el sistema final. Waterman (1986) divide estos prototipos en cinco categorías a partir de una serie de criterios: uso, robustez, eficacia, número de reglas, tiempo de desarrollo, etc. Hoy en día ya no existe una correspondencia clara entre el número de reglas y el nivel del prototipo aunque en esta división se siguen manteniendo los criterios de Waterman. Los distintos prototipos son:

1. *Prototipo de demostración.* Es un programa pequeño que contiene una parte del problema a resolver. Este programa se usa generalmente de dos formas: (a) para convencer a los potenciales usuarios y financiadores que la tecnología de los sistemas expertos puede ser empleada para resolver el

problema en cuestión, y (b) para comprobar ideas sobre la definición del problema, el ámbito y la representación del dominio. En sistemas basados en reglas estos prototipos suelen contener de 50 a 100 reglas, son capaces de resolver uno o dos casos y su desarrollo suele llevar unos tres meses.

2. *Prototipo de investigación.* Es un programa de tamaño medio y que tiene un rendimiento aceptable en una serie de casos de prueba. Estos sistemas tienden a ser frágiles y a fallar cuando se le presentan casos que están en el límite de aplicación del sistema. Debido a que la validación no ha sido intensiva también pueden fallar en casos que caen dentro de su ámbito. Un prototipo de investigación suele contener de 200 a 500 reglas, se comporta correctamente en un gran número de casos y su desarrollo suele llevar de uno a dos años.

3. *Prototipo de campo.* Son sistemas de tamaño medio y grande que han sido validados a través de casos reales. Son moderadamente fiables, contienen interfaces amigables y recogen las necesidades de los usuarios finales. Un prototipo de campo suele contener entre 500 y 1000 reglas, se comporta muy bien en un gran número de casos y su desarrollo puede llevar de 2 a 3 años.

4. *Prototipo o modelo de producción.* Son grandes sistemas que han sido intensivamente probados en el entorno de trabajo real y que pueden ser reimplementados en lenguajes más eficientes para incrementar su velocidad y reducir las necesidades de almacenamiento. Un prototipo de producción suele contener entre 500 y 1500 reglas; es correcto, rápido y eficiente en la toma de decisiones y su desarrollo lleva de 2 a 4 años.

5. *Sistema comercial.* Son prototipos de producción que son usados de forma regular en sistemas o ámbitos comerciales. XCON, uno de los mejores ejemplos de un sistema comercial tenía alrededor de 3000 reglas, alcanzaba conclusiones correctas entre un 90 y un 95 por cien de las veces y su desarrollo necesito de 6 años.

Son muy pocos sistemas los que alcanzan el prototipo de producción y menos aún los que llegan a convertirse en un sistema comercial. La mayoría de los sistemas se quedan en las fases de prototipo de demostración o prototipo de investigación.

3.4.4. Metodología en espiral

Como veíamos cuando hablábamos del software convencional, las metodologías incrementales se mejoraron con la llegada de las metodologías en espiral. Con los sistemas expertos pasó algo parecido. De esta forma, el desarrollo incremental del sistema pasó a representarse no como un bucle cuyas fases se repiten sucesivamente, sino como una espiral en la que las fases se repiten, aunque con modificaciones a medida que avanza el desarrollo.

En la bibliografía podemos encontrar muchas referencias que defienden la metodología en espiral como la más adecuada para los sistemas expertos (Lee y O'Keefe, 1994), (Lowry y Duran, 1989), (Culbert et al., 1987), (Noblett y Jones, 1991), (Cardeñosa et al., 1991), (Stachowitz y Combs, 1987), etc. De todos ellas destacaremos

el trabajo de Lee y O'Keefe ya que prestan una especial atención al problema de la validación y verificación de sistemas expertos.

Lee y O'Keefe presentan un modelo de desarrollo que sigue el modelo en espiral propuesto por Boehm (1988), pero modificado para aceptar las particularidades de los sistemas inteligentes. Este modelo lo podemos ver en la Figura 3.10.

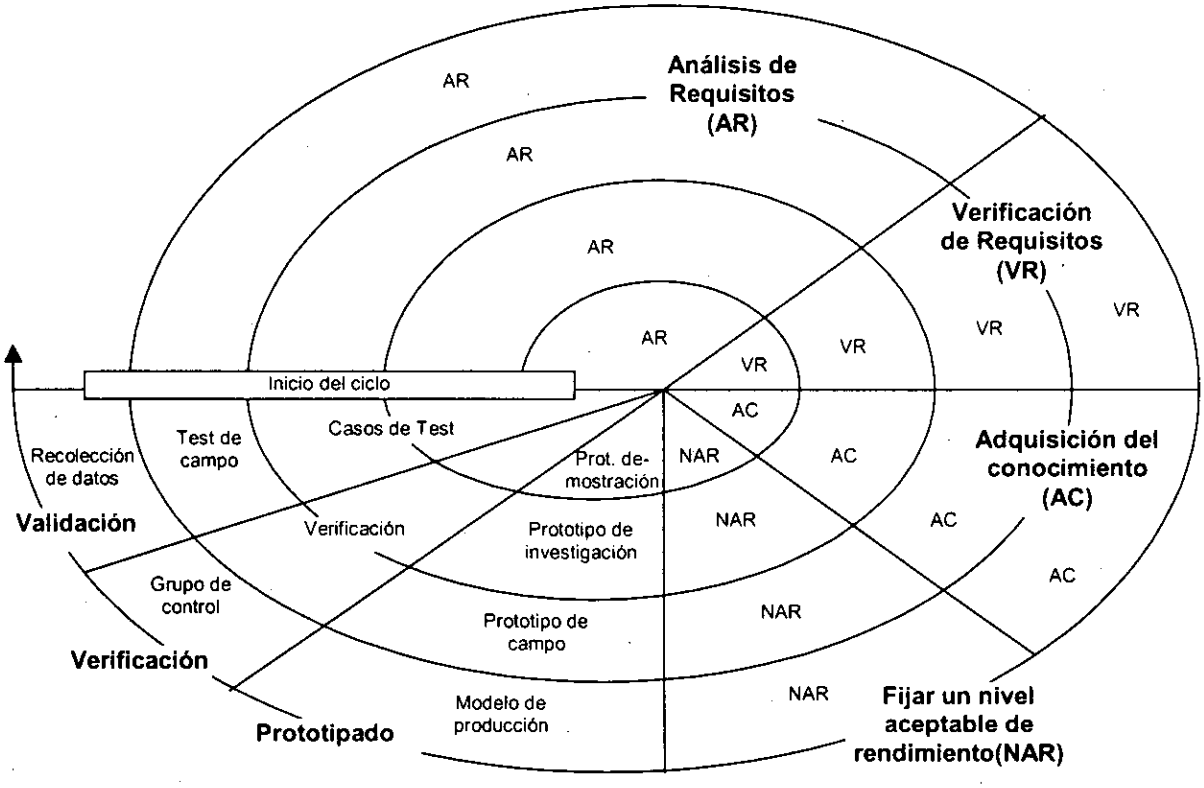


Figura 3.10. Modelo espiral de desarrollo de un sistema experto.

Lee y O'Keefe no pretenden desarrollar "el método" de construcción de sistemas expertos sino que presentan su aproximación al problema, e indican que cualquier otra solución puede ser igualmente válida siempre que incluya el prototipado rápido y el desarrollo incremental.

En este modelo es de destacar la inclusión del término "nivel aceptable de rendimiento". Esta expresión fue introducida por O'Keefe et al. (1987) haciendo referencia al hecho de que, al validar un sistema experto, no se debería clasificar su rendimiento como válido o inválido ya que, al ser los sistema expertos representaciones o abstracciones de la realidad, nunca se iba a obtener un rendimiento perfecto. En vez de eso se fija un nivel de aceptabilidad que el sistema debe cumplir y que puede ser fijado por los usuarios, los desarrolladores, los encargados de la financiación del proyecto o por terceras personas

Imaginemos un sistema que produce una interpretación dividida en 4 categorías: A, B, C y D. Un sistema ideal obtendría rendimientos de casi en 100% en su clasificación, sin embargo, un nivel aceptable de rendimiento puede no ser tan estricto. Así, por ejemplo, podemos indicar que la clasificación de A y B debe ser correcta pero se permite cierta variación en la clasificación de C y D, además nunca se debe clasificar un caso A como B, pero puede no ser tan importante si un caso B se clasifica como A.

Muchas veces el nivel aceptable de rendimiento reflejará la habilidad de trabajar a niveles similares a los de la experiencia humana (como el sistema es un modelo del experto no trabajará mejor que el experto). Así Bachant y McDermott (1984) comprobaron que al sistema R1 los usuarios no le pedían más de lo que le pedían a sus predecesores humanos.

El nivel aceptable de rendimiento se define normalmente en los requisitos del sistema pero a menudo es redefinido y expandido después de la adquisición de conocimiento. La comprobación del cumplimiento de este nivel se realiza a través de diversos métodos (pruebas basadas en casos, tests de Turing, tests de campo, etc.) que veremos en próximos apartados de este trabajo. En las ultimas espirales esta comprobación tenderá a desplazarse hacia criterios de usabilidad, siempre teniendo en cuenta el tipo de sistema que estamos construyendo.

En este modelo el desarrollo de un sistema experto se divide en cuatro fases:

1. *Análisis de requisitos.* Una respuesta fundamental que hay que responder a la hora de desarrollar un sistema es "¿es de utilidad el sistema?" (O'Keefe, 1989). Es muy importante fijar desde el inicio cuál es el problema a resolver, cuáles son los potenciales usuarios y cuál es el impacto que tendrá el sistema en la organización (un sistema que funcione perfectamente pero que no se adapte a la forma de realizar las tareas por los usuarios no tendrá ningún valor, porque no será usado). Por ello es necesaria una fase de definición de requisitos en los que se especifiquen cuestiones como: qué tipo de problemas se quieren resolver o en qué entornos se van a ejecutar, etc. Así, el problema a resolver debe ser adecuado para la aplicación de técnicas de IA, debe poder descomponerse en subproblemas de forma que los ingenieros del conocimiento puedan organizar el conocimiento, etc.

2. *Adquisición del conocimiento.* La adquisición del conocimiento, como sabemos, es un proceso que consiste en la extracción de conocimiento de una fuente de experiencia y su transformación en un esquema de representación dado.

   El conocimiento extraído debe de ser verificado. Boose (1986) propone la utilización de diagramas (grafos, tablas, jerarquías) que representen el proceso de solución del problema. Estos diagramas deben ser presentados al experto para su verificación. Como veremos en próximos apartados, muchas herramientas de construcción de sistemas expertos ya presentan la inclusión de mecanismos para la verificación automática del conocimiento.

   En esta metodología se trata de forma distinta el análisis de requisitos y la adquisición del conocimiento. Sin embargo, existen metodologías que tratan de cubrir elementos de ambas fases. Una de estas metodologías es KADS (Wielinga et al., 1992) desarrollada en Europa bajo el proyecto ESPRIT. En KADS el conocimiento es estructurado en cuatro niveles: un *nivel del dominio* que consiste en hechos básicos, conceptos y relaciones; un *nivel inferencial* que describe los procesos de resolución de problemas de una manera declarativa; el *nivel de tareas* que incluye descripciones procedimentales de las tareas que pueden llevarse a cabo

utilizando partes del nivel inferencial y; un *nivel estratégico* que suministra modelos de resolución de problemas complejos y los relaciona con el entorno. Este último nivel a menudo no ha sido considerado por los ingenieros del conocimiento, y es un problema que está en fase de investigación. Viéndolo de forma conjunta podemos decir que el nivel estratégico controla diversas tareas que aplican inferencias que utilizan hechos y relaciones del dominio.

Los autores de KADS citan que, en sus primeros intentos su metodología de adquisición seguía un modelo en cascada (Barthélemy et al., 1987), pero posteriormente se sustituyó por una metodología en espiral (Taylor et al., 1989).

3. *Prototipado.* El desarrollo incremental del sistema a través de una serie de prototipos permite que en cada ciclo se fijen los requisitos a cumplir, así como un mejor conocimiento de los objetivos del sistema y de las expectativas de los usuarios. Lee y O'Keefe presentan la estructura de prototipos propuesta por Waterman (1986).

   Sin embargo, para que la construcción de prototipos sea efectiva es necesario realizar una validación de los mismos. Para ello tenemos muchas técnicas, y la elección de la más adecuada dependerá de las características del sistema, de las características del dominio de aplicación y de la etapa de desarrollo en que nos encontremos. Así, por ejemplo, en los primeros prototipos primará la verificación automática del código, posteriormente se realizarán tests con casos de prueba y tests de Turing. Cuando el sistema esté más evolucionado se realizarán tests de campo, y cuando el prototipo constituya un modelo de producción pueden utilizarse técnicas como el *grupo de control* y recoger datos para futuras validaciones orientadas al uso. Todas estas técnicas las veremos con más detalle en apartados posteriores.

4. *Implementación y mantenimiento.* Una vez hemos desarrollado un prototipo de un sistema experto tenemos dos posibles opciones: (a) utilizar el prototipo como una fuente de especificaciones, o (b) más normalmente, evolucionar el prototipo hasta convertirlo en un sistema de producción implementado.

   Una vez el sistema esta operativo se debe monitorizar, se debe comprobar su concordancia con los requisitos, y se debe documentar la utilización del mismo en el dominio de aplicación incorporando, si fuera necesario, los nuevos requisitos de diseño que puedan surgir.

   El mantenimiento también requiere realizar tareas de validación, lo que Adrion et al. (1982) denominaban "tests regresivos". Estos tests se basan en la ejecución de casos antiguos para detectar contradicciones entre el conocimiento ya existente y el nuevo conocimiento introducido, y asegurar de esta forma la robustez del sistema. En muchos ocasiones los casos antiguos pueden no tener validez porque los límites del sistema han cambiado.

3.5. Estructura del análisis de comportamiento de un sistema experto

Ya hemos mencionado antes que un sistema experto es al mismo tiempo un software convencional y un modelo del conocimiento humano. La verificación y la validación de la parte software puede ser realizada siguiendo la metodología de la ingeniería del software, pero la parte propiamente heurística del sistema necesita técnicas particulares.

A pesar de que en la ingeniería del software términos como validación o verificación están bastante bien definidos, al intentar adaptar estos mismos términos a la ingeniería de conocimiento el consenso encontrado no es tan grande. Así, a pesar de que se han hecho intentos para unificar la terminología (Hoppe y Meseguer, 1993), lo normal es que cada autor desarrolle su propia definición de verificación y validación. A pesar de las diferencias que podamos encontrar entre las distintas definiciones siempre existe una parte común que podemos extraer de todas ellas y que intentaremos mostrar en este apartado.

En primer lugar es importante destacar que, aunque la V&V constituye la primera y más importante parte del análisis de comportamiento de un sistema experto, existen también fases posteriores. Así, el análisis de comportamiento puede verse como una pirámide basada en la verificación y validación, a partir de las cuales se desarrollan una serie de actividades que permiten asegurar la calidad del sistema. Esta representación puede verse en la Figura 3.11 en la que las fases posteriores a la V&V se han agrupado bajo el término evaluación.

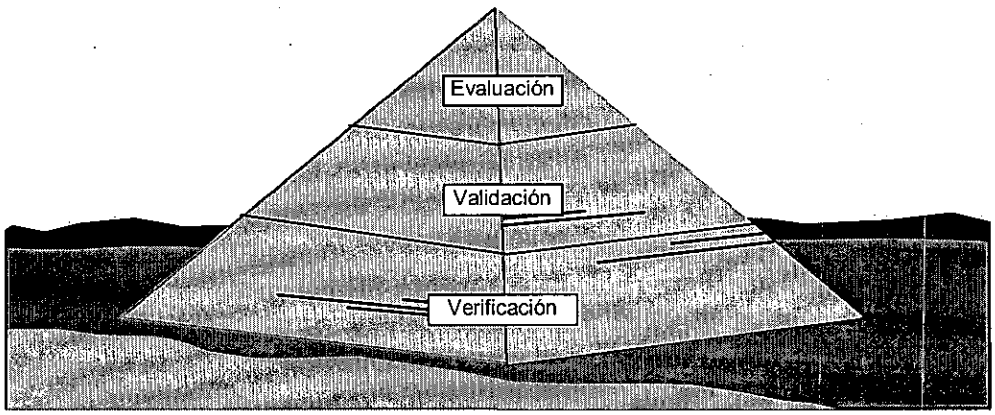


Figura 3.11 Pirámide del análisis del comportamiento de un sistema experto.

La *verificación* es, según Boehm (1981), la comprobación de que estamos construyendo el producto correctamente. A la hora de tratar sistemas expertos esta definición se formula de forma distinta, como por ejemplo "comprobar que el sistema no tiene errores y cumple sus especificaciones iniciales".

La *validación*, también según Boehm (1981), es la comprobación de que estamos construyendo el producto correcto. Lo que, expresado en términos de sistemas expertos quedaría como "comprobar que la salida del sistema sea correcta y que se cumplen las necesidades y los requisitos del usuario".

Muchos autores incluyen una o varias fases después de la validación, comúnmente agrupadas bajo el término *evaluación*, que se encargarían de analizar

aspectos que van más allá de la corrección de las soluciones finales. Así la evaluación se encargaría de analizar aspectos como utilidad, robustez, velocidad, eficiencia, posibilidades de ampliación, facilidad de manejo, etc.

La fase de evaluación es quizá la menos estudiada de las tres, principalmente porque se supone que para llegar a ella se ha realizado una verificación y una validación exitosas y el sistema está en sus últimas etapas de desarrollo. Una metodología para la realización de la evaluación es descrita en (Liebowitz, 1986).

3.6. Resumen

En este capítulo hemos descrito cuáles son las principales características de un sistema experto y cuáles son los principales problemas a los que tiene que enfrentarse la ingeniería del conocimiento (adquisición, métodos de representación y mecanismos de razonamiento). Estos problemas no aparecían, o bien aparecían de forma minimizada, en la ingeniería del software.

También se hace hincapié en las diferencias existentes entre el software convencional y los sistemas expertos, que generalmente se agrupan en las siguientes categorías: estructura, problemas apropiados, estrategias de resolución, naturaleza del conocimiento empleado y tipo de información utilizada.

A pesar de estas diferencias no es necesario partir de cero, y parte de la experiencia acumulada en el desarrollo y validación de los sistemas convencionales es exportable a los sistemas expertos. Las metodologías de construcción de los sistemas expertos se inclinan sobretodo, hacia las metodologías de desarrollo incremental, entre las que destaca de nuevo la metodología en espiral de Boehm, pero adaptada a las características específicas de la ingeniería del conocimiento.

Finalmente se detalla como se estructura el análisis del comportamiento de los sistemas expertos y en qué consisten las fases que componen dicho análisis (verificación, validación y evaluación). En los próximos capítulos veremos más en detalle las fases de verificación y validación.

4. VERIFICACIÓN DE SISTEMAS EXPERTOS

> Aristóteles afirmaba que las mujeres tenían menos dientes que los hombres, aunque estuvo casado dos veces nunca se le ocurrió verificar su afirmación examinando las bocas de sus mujeres.
> *Bertrand Russell (Filósofo y matemático británico. 1872 – 1970)*
>
> Hay tres métodos principales para adquirir conocimiento: observación de la naturaleza, reflexión y experimentación. La observación adquiere hechos, la reflexión los combina y la experimentación verifica los resultados de la combinación.
> *Denis Diderot (Enciclopedista y filósofo francés. 1713 – 1784)*

Ya hemos comentado que la verificación pretende comprobar que el sistema desarrollado cumple sus especificaciones y no contiene errores. La verificación es la fase más estudiada dentro del análisis del comportamiento de los sistemas expertos y es la más similar a su homónima dentro de la ingeniería del software. El proceso incluye las siguientes tareas: (a) verificación del cumplimiento de las especificaciones, (b) verificación de los mecanismos de inferencia, y (c) verificación de la base de conocimientos.

4.1. Verificación del cumplimiento de las especificaciones

El análisis de las especificaciones puede ser llevado a cabo por los desarrolladores, los usuarios, los expertos y/o un grupo de evaluadores independientes. En el software convencional este proceso está cada vez más automatizado con el advenimiento de las herramientas de ingeniería del software asistida por ordenador (CASE). Sin embargo, la inclusión de estas herramientas en el ámbito de la ingeniería del conocimiento es lenta.

Las cuestiones a analizar en este proceso consisten en, según Gonzalez y Dankel (1993), comprobar si:

- Se ha implementado el paradigma de representación del conocimiento adecuado.
- Se ha empleado la técnica de razonamiento adecuada.
- El diseño y la implementación han sido llevados a cabo modularmente.
- La conexión con el software externo se realiza de forma adecuada.
- El interfaz de usuario cumple las especificaciones.
- Las facilidades de explicación son apropiadas para los potenciales usuarios del sistema.
- Se cumplen los requisitos de rendimiento en tiempo real.
- El mantenimiento del sistema es posible hasta el grado especificado.
- El sistema cumple las especificaciones de seguridad.
- La base de conocimientos está protegida ante modificaciones realizadas por personal no autorizado.

La comprobación de errores en el sistema experto debe referirse a cada uno de sus componentes principales: los mecanismos de inferencia y la base de conocimientos.

4.2. Verificación de los mecanismos de inferencia

El uso de shells comerciales ha reducido la dificultad de la verificación de los mecanismos de inferencia, ya que, se asume que ésta ha sido realizada por los desarrolladores de la herramienta. La responsabilidad del ingeniero del conocimiento recae fundamentalmente en la elección de la herramienta apropiada.

Sin embargo, esta asunción de correcto funcionamiento no siempre es cierta (sobre todo en versiones nuevas de la herramienta). Por ello, para aplicaciones que trabajan en dominios críticos, el funcionamiento correcto debe verificarse a través de distintas pruebas.

Muchas veces los problemas con las shells comerciales pueden no ser causa de errores en su programación. Así, en ocasiones hay que pensar en un desconocimiento del funcionamiento exacto de la herramienta. Por ejemplo, los procedimientos de resolución de conflictos o los mecanismos de herencia pueden hacer difícil el seguimiento del curso exacto de la inferencia. De esta forma, aunque el conocimiento estático esté verificado, el funcionamiento final del sistema puede no ser el apropiado.

En caso de que decidamos construir nuestros propios mecanismos de inferencia, será preciso realizar su verificación. Como estamos tratando software convencional podemos aplicar para su verificación las técnicas vistas cuando hablábamos de la verificación dentro de la ingeniería del software.

Geissman y Schultz (1988) recomiendan la utilización, siempre que sea posible, de mecanismos de inferencia certificados, es decir, cuyo funcionamiento correcto se haya probado. Además en caso de utilizar herramientas comerciales aconsejan realizar pruebas para comprobar que realmente se comportan como indican en sus manuales.

4.3. Verificación de la base de conocimientos.

La verificación de la base de conocimientos, a diferencia de los mecanismos de inferencia, es plena responsabilidad del ingeniero del conocimiento. Esta verificación se basa, generalmente, en el concepto de *anomalías*. Una anomalía es un uso poco común del esquema de representación del conocimiento, que puede ser considerado como un error potencial (existen anomalías que no constituyen errores y viceversa).

La verificación de la base de conocimientos no nos asegura que las respuestas de nuestro sistema sean correctas, lo que nos asegura es que el sistema ha sido diseñado e implementado de forma correcta.

La mayoría de los estudios publicados que tratan sobre la verificación de las bases de conocimientos se refieren a los sistemas basados en reglas, ya que son los más populares. Por ello en este estudio nos centraremos en dichos sistemas. Esto no quiere decir que los sistemas construidos según otros paradigmas no necesiten ser verificados o que su verificación no sea posible. Así, por ejemplo, Cheng (1989) muestra como se llevaría a cabo la verificación de un sistema experto basado en frames; Shiu et al. (1997) realizan una verificación formal de un sistema que utiliza reglas y frames; y Kandelin y O´Leary (1995) realizan la verificación de un sistema orientado a objetos.

Aspectos que se suelen examinar a la hora de verificar una base de conocimientos son la consistencia y la completitud. A continuación veremos una serie de pruebas que se realizan para comprobar que la base de conocimientos es consistente y completa. En principio supondremos que los sistemas no manejan incertidumbre, luego veremos como la inclusión de incertidumbre puede afectar a las pruebas desarrolladas.

4.3.1. Verificación de la consistencia

La verificación de la consistencia de la base de conocimientos consiste en la detección de reglas redundantes, reglas conflictivas, reglas englobadas en otras, reglas circulares y condiciones IF innecesarias. Veamos con un poco más de detalle estos posibles errores.

Reglas redundantes

Existen dos tipos de redundancias, por un lado *redundancias sintácticas* que ocurren cuando dos reglas tienen las mismas premisas y alcanzan idénticas conclusiones:

$$p(x) \wedge q(x) \rightarrow r(x)$$

$$q(x) \wedge p(x) \rightarrow r(x)$$

Por otro lado tenemos *redundancias semánticas* que ocurren cuando las premisas o las conclusiones de una regla no son idénticas en la sintaxis, pero sí en el significado.

$$p(x) \wedge q(x) \rightarrow r(x) = \text{Tormenta}$$

$$q(x) \wedge p(x) \rightarrow r'(x) = \text{Actividad Eléctrica}$$

Las redundancias semánticas son menos comunes pero más difíciles de detectar ya que el sistema no advierte que el significado es el mismo a pesar de tener distinta sintaxis.

Las redundancias no causan necesariamente problemas lógicos, aunque pueden afectar a la eficiencia del sistema (Suwa et al., 1982). Sin embargo, los problemas pueden aparecer cuando, en futuras versiones del sistema, se cambie una regla pero no la otra.

Reglas conflictivas

Dos reglas son conflictivas cuando sus premisas son idénticas pero sus conclusiones son contradictorias.

$$p(x) \wedge q(x) \rightarrow r(x)$$

$$p(x) \wedge q(x) \rightarrow \neg r(x)$$

No siempre que las premisas de dos reglas sean idénticas podemos decir que ha ocurrido un conflicto. Así, por ejemplo, si la conclusión es un atributo multivaluado, podemos estar estableciendo la probabilidad de aparición de las distintas hipótesis. También puede suceder que el atributo esté tomando a la vez varios valores (i.e. una persona puede ser alérgica a varias substancias o haber sido infectada por varios organismos).

Reglas englobadas en otras

Una regla está englobada en otra si las dos reglas tienen las mismas conclusiones, pero una de ellas tiene restricciones adicionales en la premisa.

$$p(x) \wedge q(x) \rightarrow r(x)$$

$$p(x) \rightarrow r(x)$$

En este caso la regla con más restricciones en la premisa está englobada dentro de la que tiene menos. En las estrategias de resolución de conflictos entre las reglas se podría intentar ejecutar primero la regla más específica, y en caso de no ser posible, ejecutar la regla más general.

Reglas circulares

Un conjunto de reglas es circular si el encadenamiento de las mismas forma un ciclo, es decir, se comienza por una determinada condición y que al final del razonamiento volvemos de nuevo a la misma condición.

$$p(x) \rightarrow q(x)$$

$$q(x) \rightarrow r(x)$$

$$r(x) \rightarrow p(x)$$

Los conjuntos de reglas circulares pueden provocar que el sistema caiga en un bucle infinito durante su ejecución, a no ser que incluya algún mecanismo para evitar dichos bucles.

Condiciones IF innecesarias

Una condición IF innecesaria existe cuando dos reglas tienen idénticas conclusiones, una premisa de una regla está en contradicción con una premisa en la otra regla y el resto de premisas son equivalentes.

$$p(x) \wedge q(x) \rightarrow r(x)$$

$$p(x) \wedge \neg q(x) \rightarrow r(x)$$

En este caso las dos reglas pueden resumirse en una sólo conteniendo únicamente las premisas equivalentes.

$$p(x) \rightarrow r(x)$$

Puede haber situaciones especiales en la que una condición IF innecesaria no implica necesariamente que se unan las dos reglas. Así por ejemplo el conjunto de reglas

$$p(x) \wedge q(x) \rightarrow r(x)$$

$$\neg q(x) \rightarrow r(x)$$

quedaría reducido eliminando la condición IF innecesaria de la primera regla pero manteniendo intacta la segunda regla:

$$p(x) \rightarrow r(x)$$

$$\neg q(x) \rightarrow r(x)$$

4.3.2. Verificación de la completitud

Muy a menudo, el proceso de adquisición de conocimiento a partir de fuentes de experiencia no es completo, lo que puede producir "huecos" en el conocimiento adquirido. Existen una serie de situaciones típicas que pueden ser indicativas de un "hueco" en la base de conocimientos como son valores no referenciados de atributos, valores ilegales de atributos, reglas inalcanzables o reglas "sin salida". Veamos estas situaciones más en detalle.

Valores no referenciados de atributos

Esta situación ocurre cuando algunos valores, del conjunto de posibles valores de un atributo, no son cubiertos por la parte IF de ninguna regla. Por ejemplo, si tenemos el atributo Temperatura cuyo rango de posibles valores es {"alta", "normal", "baja"} y los valores "alta" y "normal" aparecen en la parte IF de alguna regla, pero no ocurre así con el valor "baja".

Un atributo parcialmente cubierto puede impedir que el sistema alcance una conclusión o puede provocar conclusiones erróneas cuando el valor no cubierto aparece en al ejecución. Este error puede indicar la falta de alguna regla en la base de conocimientos.

Valores ilegales de atributos

Esta situación ocurre cuando una regla referencia valores de atributos que no están incluidos en el conjunto de valores válidos para ese atributo. Por ejemplo, si el conjunto de valores válidos de Temperatura es {"alta", "normal", "baja"} y encontramos condiciones del tipo Temperatura = "muy alta" o Temperatura = "algo baja", tanto en premisas como en conclusiones.

Este error es causado, generalmente, por equivocaciones en la escritura, aunque también puede ser indicativo de que el conjunto de valores válidos del atributo es incompleto.

Reglas inalcanzables

En un sistema de producción que utiliza una búsqueda regresiva, una regla es inalcanzable si la conclusión de la regla no aparece en el objetivo a buscar y no aparece en la condición IF de otra regla. Por ejemplo la regla "IF Temperatura > 37 THEN Fiebre" sería inalcanzable si "Fiebre" no aparece en el objetivo buscado ni en la condición IF de otra regla.

En caso de que el razonamiento fuese progresivo, la regla sería inalcanzable si sus premisas no pueden ser obtenidas del exterior (por ejemplo, preguntándole al usuario) y no aparecen como conclusión de ninguna regla. Siguiendo con el ejemplo anterior, la regla "IF Temperatura > 37 THEN Fiebre" sería inalcanzable si el valor de "Temperatura" no puede ser actualizado desde el exterior, ni aparece como conclusión de alguna otra regla.

Esta situación puede afectar a la eficiencia del sistema pero nunca a su salida (ya que la regla con la conclusión inalcanzable nunca será disparada). Una causa muy común de este tipo de situaciones son errores en la terminología. Así, por ejemplo, la conclusión "Fiebre" podría aparecer en la parte IF de una regla de la siguiente forma:

IF Temperatura_Elevada THEN ...

en donde los términos "Temperatura_Elevada" y "Fiebre" son sinonimos para el experto, pero no para el sistema experto.

Reglas "sin salida"

Esta situación es la inversa a la vista en el punto anterior. Una regla inalcanzable para un razonamiento regresivo es una regla "sin salida" para un razonamiento progresivo. De la misma forma una regla inalcanzable para un razonamiento progresivo es una regla "sin salida" para un razonamiento regresivo.

Así, si la regla "IF Temperatura > 37 THEN Fiebre" era inalcanzable para un razonamiento regresivo porque "Fiebre" no aparecía ni en la conclusión, ni en la parte IF de otra regla. Si buscamos de forma progresiva, la línea de razonamiento llegaría a un punto sin salida al ejecutar esta regla ya que, después de concluir "Fiebre" no existen posibilidades para continuar con dicha línea razonamiento.

4.3.3. Influencia de las medidas de incertidumbre

Las reglas vistas hasta ahora para verificar la consistencia y la completitud son válidas siempre y cuando los sistemas no incluyan incertidumbre. En caso de que si exista dicha incertidumbre la validez de las pruebas queda en entredicho, ya que como veremos situaciones normales pueden ser tomadas como errores.

En sistemas que pretenden medir incertidumbres o grados de asociación (utilizando factores de certeza, probabilidades bayesianas o cualquier otro método) es importante verificar que estos valores son consistentes, completos, correctos y no redundantes. Esta tarea se realiza, en primer lugar, asegurándonos de que cada regla incluye un factor de incertidumbre, y que estos factores cumplen los aspectos de la teoría en la que se basan.

La búsqueda de anomalías en los factores de un sistema experto es un proceso que no ha recibido mucha atención por parte de los investigadores, quizá debido al limitado número de sistemas expertos que hacen un uso extensivo de las medidas de incertidumbre. Un ejemplo de verificación de este tipo es el trabajo que O'Leary (1990) desarrolló en sistemas que seguían el esquema bayesiano.

El modo en que el uso de medidas de incertidumbre también puede afectar a la realización de los tests de consistencia y completitud puede verse en los siguientes ejemplos (Nguyen et al., 1987):

- *Redundancia*: Si antes la redundancia no afectaba a la salida del sistema ahora puede causar graves problema ya que, al contar la misma información dos veces, su pueden modificar los pesos de las conclusiones.

- *Reglas englobadas en otras*: Esta situación puede no ser errónea ya que las dos reglas pueden indicar la misma conclusión pero con distintas confianzas. La regla englobada sería un refinamiento de la regla más general para el caso de que tengamos más información.

- *Condiciones IF innecesarias*: Igual que en el caso anterior, una condición IF innecesaria puede utilizarse para variar la confianza en la conclusión final.

- *Reglas circulares*: Pueden existir casos en los que la utilización de medidas de incertidumbre rompan la circularidad de un conjunto de reglas. Por ejemplo, si el factor de certidumbre de una conclusión implicada en el ciclo cae por debajo de un umbral (normalmente entre –0.2 y 0.2) se considera que el valor de la conclusión es "desconocido" y el ciclo se rompe.

- *Reglas "sin salida"*: La detección de este tipo de reglas se complica con la introducción de incertidumbre. Así, una regla puede convertirse en una regla "sin salida" si su conclusión tiene una certidumbre por debajo del umbral en el cual un valor se considera "conocido". Por ejemplo, la siguiente cadena de reglas

$$A \xrightarrow[0.4]{R1} B \xrightarrow[0.7]{R2} C \xrightarrow[0.7]{R3} D$$

  podría parecer válida, sin embargo si A se conoce con total certidumbre, el factor de certeza de D después de un razonamiento progresivo sería $0.4 \times 0.7 \times 0.7 = 0.196$ (menor que 0.2) con lo que el valor de D sería "desconocido" y la línea de razonamiento acabaría en un punto "sin salida".

- *Reglas inalcanzables*: de forma similar al ejemplo anterior pueden existir reglas que, por causa de los factores de certeza, se convierten en inalcanzables. Si consideramos el siguiente conjunto de reglas

$$A \xrightarrow[0.1]{R1} B \xrightarrow[1]{R2} C$$

la regla R2 sería inalcanzable en un razonamiento progresivo (aunque su premisa aparece en la conclusión de otra regla) porque el valor de B cae por debajo del umbral de 0.2.

4.4. Verificación dependiente o independiente del dominio

El tipo de verificación que hemos visto hasta ahora se denomina *independiente del dominio* porque no es específica de ningún dominio en particular. Se basa, como hemos visto, en una detección de anomalías que pueden ser errores o no. Esta búsqueda de anomalías se realiza normalmente a través de aproximaciones heurísticas.

Sin embargo, existe un tipo de verificación que es *dependiente del dominio* (O'Keefe y O'Leary, 1993). Este tipo de verificación emplea metaconocimiento del dominio para examinar la base de conocimientos. El metaconocimiento se define como el conocimiento que el sistema tiene acerca de su propia estructura.

El primer ejemplo, y también el mejor conocido, de esta técnica aparece con los trabajos de Davies (1976) en TEIRESIAS, un sistema que se encargaba de verificar la introducción de conocimiento nuevo en el sistema experto MYCIN.

Un ejemplo de la verificación dependiente del dominio sería la siguiente: nuestro metaconocimiento nos puede indicar las características del salón de una casa. El proceso de verificación puede detectar que el salón está incompleto si no hemos puesto un sofá, pueden existir redundancias si hemos puesto más de un sofá, puede haber incorrecciones si hemos puesto el lavabo en la sala y pueden existir inconsistencias si ponemos un sofá y una silla, y a ambos los denominamos "sofá". Como vemos, todo el proceso de validación se basa en el conocimiento previo que tenemos sobre como debería ser un salón.

La verificación independiente del dominio se encuentra, frecuentemente, incluida en los procesos y herramientas de adquisición del conocimientos (recordar que TEIRESIAS se incluía en el capítulo 3.2.1 como un programa inteligente de edición para la adquisición de conocimiento, otros ejemplos aparecen en (Boose y Bradshaw, 1987) y (Gaines, 1987)).

Sin embargo, a pesar de las ventajas evidentes que supone utilizar metaconocimiento en la verificación, existen una serie de inconvenientes que limitan su aplicación práctica:

- El metaconocimiento tiene que ser verificado (ya que también es conocimiento).
- El metaconocimiento puede no ser estable, o puede no ser práctico actualizarlo con frecuencia.
- En general, el desarrollo de una aproximación dependiente del dominio puede ser bastante costosa debido a los costes de adquirir y mantener el metaconocimiento.

Por estas razones la mayoría de los sistemas se verifican con una aproximación independiente del dominio.

4.5. Herramientas

De las distintas fases que componen el análisis del comportamiento de un sistema experto, la fase de verificación es en la que ha conseguido un mayor grado de automatización mediante distintos tipos de herramientas. Dentro de estas herramientas de verificación podemos establecer dos grupos: las dependientes y las independientes del dominio.

4.5.1. Herramientas dependientes del dominio

Dentro de las herramientas dependientes del dominio destacamos el ya nombrado TEIRESIAS (Davies, 1976). Esta herramienta era capaz de identificar errores en la base de conocimientos y corregirlos mediante la modificación, adición o borrado de reglas. En versiones posteriores TEIRESIAS era capaz de corregir interacciones incorrectas entre las reglas, comprobar la sintáctica y la semántica, generar explicaciones de los sistemas de razonamiento y comparar los resultados del sistema contra las conclusiones alcanzadas por un experto del dominio. Esto hacía de TEIRESIAS una buena herramienta para el desarrollo incremental de la base de conocimientos. Sin embargo tenía el inconveniente de que requería que los mecanismos de razonamiento fueran funcionales antes de empezar con los tests.

Otra herramienta dependiente del dominio es EVA (Expert systems Validation Associate) desarrollada por Stachowitz y Combs (1987). EVA interactúa con shells estándar de sistemas expertos, como KEE, de forma que los hechos y las reglas son trasladados al formato de EVA, el cual se encarga de verificarlos utilizando un algoritmo de chequeo de estructuras y el metaconocimiento previamente desarrollado. Así, por ejemplo, si tenemos las reglas

$$R1: HUMANO(x) \rightarrow MORTAL(x)$$

$$R2: PERSONA(x) \rightarrow MORTAL(x)$$

y el metaconocimiento

$$M1: SINÓNIMO(HUMANO, PERSONA)$$

EVA detectaría las reglas R1 y R2 son equivalente por lo que una de ellas podría ser eliminada.

Uno de los principales problemas de EVA es el control del la base de metaconocimientos, ya que puede darse el caso de que su tamaño sea incluso mayor que el de la propia base de conocimientos.

4.5.2. Herramientas independientes del dominio.

La mayoría de las herramientas de verificación existentes son independientes del domino, ya que analizan la estructura de la base de conocimientos sin tener en cuenta el

dominio en el que estamos trabajando. Estas herramientas suelen convertir la base de conocimientos en una representación independiente (mediante tablas o grafos) a partir de la cual se buscan las posibles anomalías.

Entre los primeros trabajos de verificación independiente del dominio podemos destacar el Rule Checker Program (RPC) desarrollado por Suwa et al. (1982) como un asistente para la verificación de la base de conocimientos de ONCOCIN (un sistema experto sobre medicina oncológica). Una de las principales contribuciones de este programa es que permite realizar la verificación a medida que el sistema experto se va desarrollando. RPC analiza la base de conocimientos a partir de una tabla de decisión en la que se muestran todas las posibles combinaciones de valores que pueden tomar los atributos de condición y los relaciona con los correspondientes valores que concluiría el sistema. La herramienta ESC (Expert System Checker) de Cragun y Steudel (1987) también utiliza tablas de decisión.

Otro trabajo importante es el desarrollado por Nguyen et al. (1987) con la herramienta CHECK. Esta herramienta está preparada para trabajar con la shell LES (Lockheed Expert System) y es una extensión del trabajo realizado por Suwa et al. (1982). CHECK incluye muchos criterios para la validación de las reglas y realiza tablas y gráficos de dependencias para mostrar las interrelaciones entre las distintas reglas.

Un ejemplo de tablas y gráficos de dependencia puede verse en la Figura 4.1. En ella tenemos en primer lugar una serie de reglas iniciales (sin ciclos) y una meta. Si variamos ligeramente las reglas 2 y 3 vemos que aparecen una serie de dependencias que indican la presencia de ciclos (marcadas con $*^2$ y $*^3$). Estos ciclos se detectan construyendo un grafo a partir de la tabla de dependencia que representa la interacciones entre las reglas y detectando ciclos en dicho grafo.

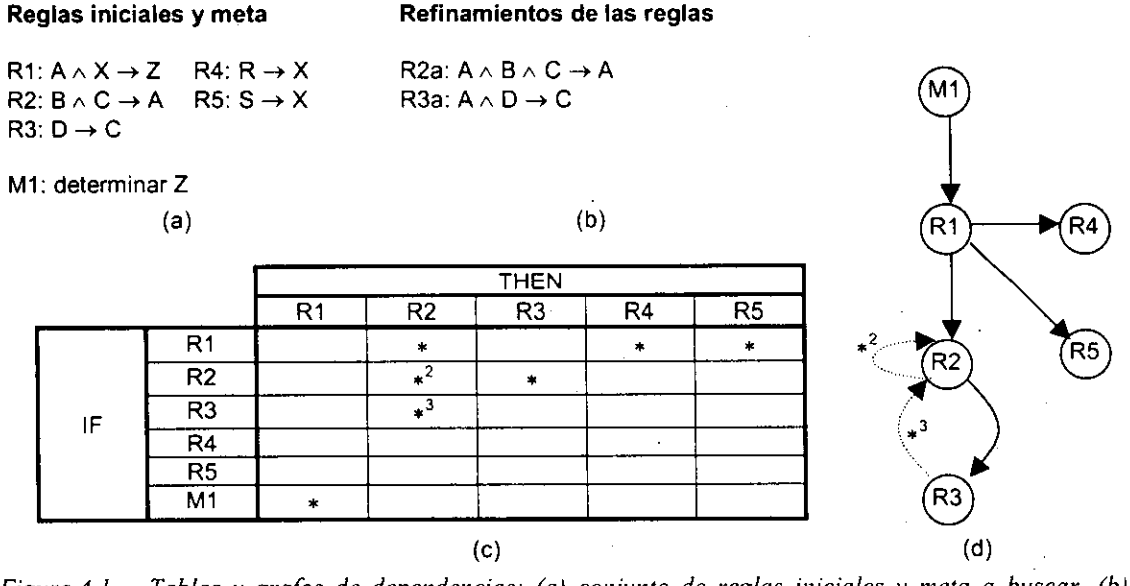


Figura 4.1 *Tablas y grafos de dependencias: (a) conjunto de reglas iniciales y meta a buscar, (b) refinamientos posteriores de las reglas R2 y R3, (c) tabla de dependencias ($*^2$ y $*^3$ son el resultado de refinar las reglas R2 y R3), (d) grafo de dependencias mostrando en punteado los ciclos que surgen al modificar las reglas R2 y R3.*

Como mejora a los métodos basados en tablas tenemos la herramienta COVER (Preece et al., 1992). Esta herramienta construye, directamente de las reglas, un grafo

que las representa. La ventaja de esta técnica es que permite detectar anomalías entre numerosas reglas, y no sólo entre pares de reglas como es común en las aproximaciones basadas en tablas.

Otro avance lo representa la herramienta KB-Reducer (Ginsberg, 1988). En esta herramienta las reglas son transformadas en una representación basada en la lógica y para cada hipótesis se describe una *etiqueta* que contiene las condiciones bajo las cuales la hipótesis considerada es cierta. A medida que se van generando las distintas etiquetas se comprueba la redundancia, las contradicciones y la inconsistencia del sistema. Teóricamente, KB-Reducer puede detectar todas las potenciales contradicciones en una base de reglas (basada en objetos, atributos y valores) para un mecanismo de inferencia que cumpla: (1) es monótono, (2) no utiliza una estrategia de resolución de conflictos, y (3) es dirigido por los datos, en el sentido en que todos los datos requeridos están en la memoria de trabajo.

En un intento de realizar menos supuestos en cuanto a los mecanismos de inferencia y de resolución de conflictos, aparecen una serie de herramientas que hacen uso de las redes de Petri. Dentro de esta técnica incluimos los trabajos de Shiu et al. (1997) y de Wu y Lee (1997) como artículos más recientes; también son comúnmente referenciados los trabajos de Agarwal y Tanniru (1992), Liu y Dillon (1991) y Pipard (1988). Una de las ventajas del empleo de las redes de Petri es que permite analizar las relaciones temporales entre las reglas, sin embargo, tienen el inconveniente de que la conversión de una base de reglas a una red de Petri no es una tarea trivial. Una introducción sobre las redes de Petri en la informática puede encontrarse en (Silva, 85).

Otra herramienta a destacar sería Validator, de Kang y Bahill (1990) que emplea medidas estadísticas para encontrar errores en la base de conocimientos después de la ejecución de una serie de casos de prueba. Así, por ejemplo, se registran las ejecuciones de las distintas reglas y se detecta si existe alguna una regla que no se haya ejecutado nunca (o alguna que se haya ejecutado siempre) para todos los casos del tests. Este tipo de reglas probablemente sean errores.

El principal problema que se le achaca a las herramientas de verificación de bases de conocimientos, es que suponen una serie de condiciones muy simplificadas para que su funcionamiento sea efectivo, como por ejemplo:

- *Tamaño reducido*: Los métodos descritos por las distintas herramientas suelen ser adecuados cuando el número de reglas no es muy elevado. Al aumentar el número de reglas puede producirse una explosión combinatoria que reduciría drásticamente la eficiencia de la herramienta. Ginsberg destacó que su herramienta, KB-Reducer, pasaba de tardar 40 sg. con una base de conocimientos de 50 reglas a tardar 10 horas con otra de 370 reglas. Para evitar este problema se han desarrollado técnicas como la partición de la base de reglas, como se hizo en ESC, o el uso de heurísticas, como en la herramienta COVER.

- *Bases de reglas no estructuradas*: La mayoría de las herramientas de verificación suponen que la base de reglas es *plana* y que la ejecución de las reglas se hace mediante una estrategia progresiva o regresiva pero sin tener en cuenta otras estructuras de control. Sin embargo, los dominios complejos y poco estructurados en los que se pretenden desarrollar sistemas expertos

suelen obligar a la inclusión de sofisticadas capacidades de control que, la mayoría de las veces, no son contempladas por ninguna herramienta de verificación.

- *No inclusión de la incertidumbre*: Como hemos visto anteriormente la inclusión de medidas de incertidumbre puede provocar que, situaciones que antes considerábamos anómalas, ahora sean perfectamente válidas. Sin embargo son pocos los trabajos que se encargan de verificar bases de conocimientos en presencia de incertidumbre. Entre ellos podemos destacar el trabajo de Wilkins y Buchanan (1986)

Para más información sobre las herramientas de verificación y su comparación se pueden consultar los trabajos de Murrell y Plant (1997), Gupta (1993) y López et al. (1990).

4.6. Resumen

En este capítulo se ha realizado una pequeña introducción a la verificación de los sistemas expertos. La fase de verificación trata de comprobar que el sistema se ha desarrollado correctamente desde tres puntos de vista: verificar el cumplimiento de las especificaciones, verificar los mecanismos de razonamiento y verificar la base de conocimientos.

La verificación del cumplimiento de las especificaciones incluye la comprobación de si el paradigma de representación del conocimiento es adecuado, si la técnica de razonamiento es adecuada, si el diseño y la implementación han sido llevados a cabo modularmente, etc.

La verificación de los mecanismos de inferencia ha reducido su dificultad en los últimos años con el empleo de shells comerciales y motores de inferencia certificados. Esto hace que la importancia de la fase de verificación recaiga sobre la base de conocimientos.

Existen muchas técnicas para comprobar que la consistencia y la completitud de una base de conocimientos (sobre todo aquellas basadas en reglas). Estas técnicas se han implementado en muchas herramientas (dependientes o independientes del dominio), pero siguen teniendo el inconveniente de necesitar que la estructura de la base de conocimientos no sea demasiado compleja.

Después de esta introducción a la fase de verificación, el tema siguiente detalla de forma más exhaustiva la fase de validación de los sistemas expertos.

5. ASPECTOS GENERALES DE LA VALIDACIÓN DE SISTEMAS EXPERTOS

> Los errores de los arquitectos se tapan con flores.
> los de los cocineros, con salsas.
> Los de los médicos..., con tierra.
> Anónimo

Ya hemos mencionado que la validación consiste en comprobar si estamos construyendo el producto correcto; es decir, comprobamos si los resultados del sistema son correctos, y si se cumplen las necesidades y los requisitos del usuario.

La comprobación sobre la validez de los resultados del sistema se denomina *validación orientada a los resultados* (Lee y O'Keefe, 1994). Su objetivo es comparar el rendimiento del sistema con un rendimiento esperado (proporcionado por una referencia estándar o por expertos humanos), y comprobar que el sistema alcanza un nivel que se considera aceptable.

Por otro lado la *validación orientada al uso*, se centra en cuestiones que hacen referencia a la relación hombre-máquina, más allá de la corrección de los resultados obtenidos por el sistema. Este tipo de validación se suele referenciar en la literatura con el término *assessment* (O'Leary, 1987), que en castellano podríamos traducir como *valoración*.

En la validación orientada al uso se analizan aspectos como la facilidad de utilización del sistema, la calidad del diálogo hombre-máquina, la calidad de la implementación, la adecuación y la eficiencia del hardware, etc. Podemos encontrar listas de otros posibles criterios en (Riedel y Pitz, 1986) o (Adelman, 1991a).

Algunos autores, como Liebowitz (1986), no consideran esta división en la validación, sino que presuponen que la fase de validación se orienta únicamente hacia la corrección de los resultados. En este caso la validación orientada al uso se incluiría dentro de la fase de evaluación. Liebowitz también propone una metodología denominada AHP (Analytical Hierarchy Process) para realizar la validación orientada al uso.

Normalmente la validación orientada a los resultados constituye un prerrequisito para la realización de una validación orientada al uso. Así, si un sistema no presenta un rendimiento aceptable (o al menos indicaciones de que el rendimiento mejorará en un futuro al incluir mejoras en el desarrollo), los aspectos concernientes a la validación orientada al uso son irrelevantes. Por este motivo en este trabajo nos centramos básicamente en el estudio de la validación orientada a los resultados a la que, a partir de ahora, denominaremos simplemente como "validación".

Antes de describir una metodología determinada es necesario describir cuales son las características principales del proceso de validación. Al respecto destacaremos las siguientes:

- Personal involucrado en la validación.
- Partes del sistema a validar.
- Datos utilizados en la validación.

- Criterios de validación.
- Momento en el que se realiza la validación
- Métodos de validación
- Errores cometidos en la validación.

5.1. Personal involucrado en la validación

Una cuestión importante a determinar en todo tipo de validación es quién va a llevarla a cabo (Figura 5.1). El primer elemento a considerar es el ingeniero de conocimiento que ha desarrollado el sistema, ya que es quien mejor conoce las características del sistema experto. Sin embargo, incluir al ingeniero del conocimiento en el proceso puede afectar a la objetividad del mismo (ha dedicado mucho esfuerzo en el desarrollo del sistema y puede sentirse inclinado a sobrevalorar los resultados del mismo). De todas formas, en la validación siempre es necesaria la presencia de una persona que tenga un conocimiento amplio del sistema, aunque no sea su constructor.

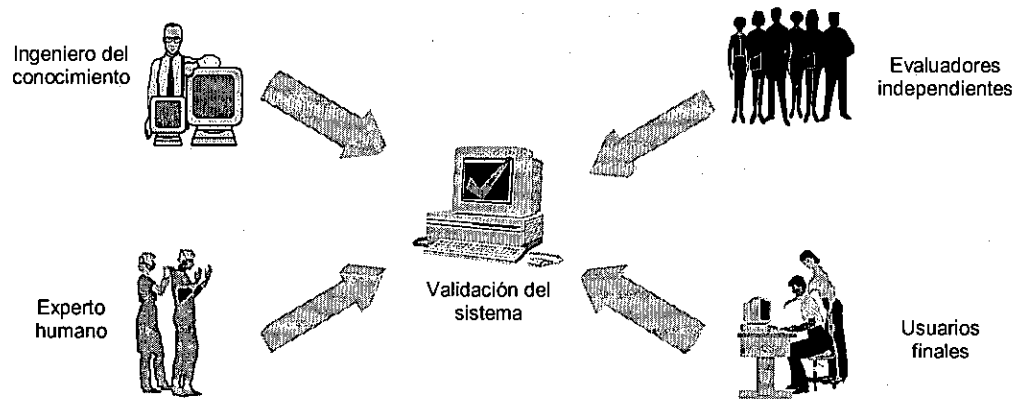


Figura 5.1. Personal involucrado en la validación de un sistema experto.

También es necesario contar con expertos humanos. Como veremos, el método básico para realizar la validación es el análisis de casos de prueba ya resueltos. Estos casos habrán sido analizados también por expertos humanos con los que podremos estudiar las discrepancias encontradas. Generalmente es conveniente que los expertos que participen en la validación no sean los mismos que colaboran en el desarrollo del sistema. Con esta medida se intenta conseguir que el conocimiento del sistema se adecúe al de un consenso de expertos (y no sólo al conocimiento del experto colaborador en el diseño). No obstante, el tiempo de los expertos humanos es muy valioso, por lo que puede ser complicado contar con un amplio número de ellos para realizar una validación amplia y exhaustiva.

Debido a la necesidad de independencia en la validación surgió la idea de hacer recaer todas las responsabilidades en un grupo de expertos independiente (que O'Keefe y O'Leary (1993) denominan "terceros expertos"). Sin embargo, si el constructor del sistema podía sobrevalorar el mismo, el uso de una grupo de validación totalmente independiente puede provocar el efecto contrario (Buchanan y Shortliffe, 1984). Esta situación es la que Chandrasekaran (1983) describió como la "falacia del superhombre": se le exige más al sistema experto de lo que se le exigiría a un experto humano (teniendo en cuenta que el conocimiento del sistema experto es simplemente un modelo del conocimiento de los expertos humanos). También pueden aparecer problemas si los

evaluadores no aceptan fácilmente la utilización de sistemas expertos en su área de trabajo, o si la solución propuesta pertenece a una "escuela de pensamiento" diferente a la suya. Como veremos posteriormente, para evitar subjetividades en el proceso de validación, se pueden llevar a cabo lo que se denominan "estudios ciegos".

Los usuarios finales del sistema también pueden participar en el proceso de validación; sin embargo, puede ocurrir que la experiencia de los mismos no sea suficiente para realizar la validación del sistema experto. Por ello generalmente su labor se destina a una validación orientada al uso.

5.2. Partes del sistema a validar

Nuestro principal objetivo es lograr que los resultados finales del sistema experto sean correctos. Sin embargo, también puede ser interesante analizar si los resultados intermedios del sistema son correctos o si el razonamiento seguido hasta dar con la solución es apropiado.

La validación de los resultados intermedios puede ser interesante porque los resultados finales dependen de ellos. Así, el análisis de dichos resultados intermedios nos da una descripción del funcionamiento interno del sistema y nos permite una rápida corrección de los errores cometidos.

También puede resultar apropiado validar las estructuras de razonamiento; es decir, comprobar que el sistema alcanza la respuesta correcta por las razones correctas (Gaschnig et al., 1983). Un proceso de razonamiento incorrecto puede provocar errores cuando queramos ampliar nuestra base de conocimientos (Chandrasekaran, 1983). En este caso lo que se pretende es emular el proceso de razonamiento que realizan los expertos humanos. De esta forma los usuarios del sistema encontrarán más agradable su utilización al seguir una línea lógica a la hora de plantear las cuestiones.

Para ver estas cuestiones con más claridad consideremos el siguiente ejemplo: sea un paciente en una unidad de cuidados intensivos, en la cual estamos monitorizando constantemente sus datos gasométricos. Además contamos con las características del contexto particular de su caso. Con estos datos intentamos hallar el estado de su balance ácido-base a través de un sistema experto. En la Figura 5.2 vemos que, ante un caso determinado, se ha producido un error y el resultado no es el esperado.

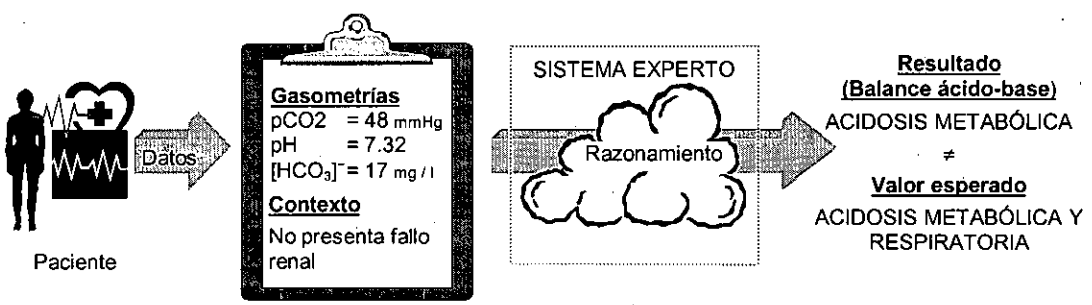


Figura 5.2. Las conclusiones finales del sistema sobre el "Balance ácido-base" no son correctas.

Analizando los resultados intermedios vemos que el error era debido a un fallo en la interpretación del pCO_2 debido a que había una errata en una de las reglas que determina el estado del pCO_2 (Figura 5.3).

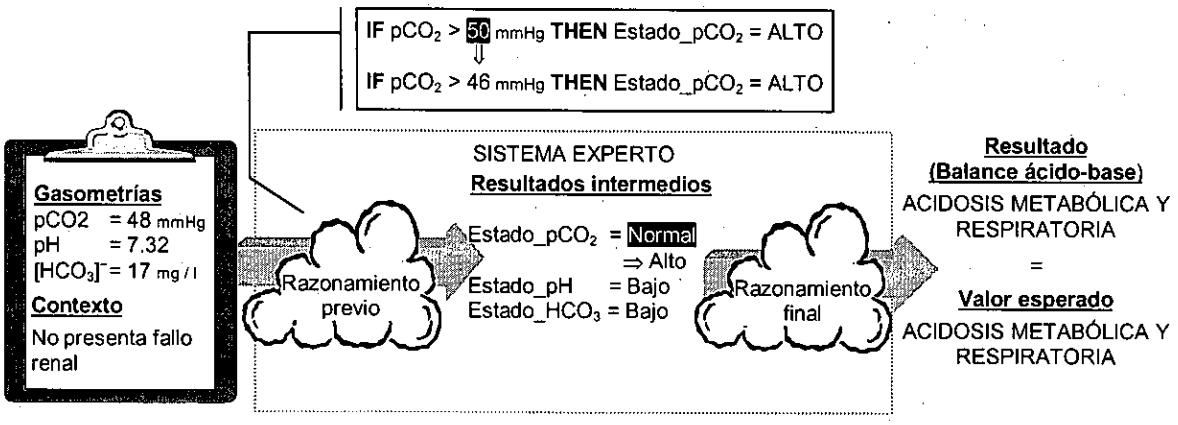


Figura 5.3. El error en la conclusiones finales era debido a que uno de los resultados intermedios (Estado_PCO2) se interpretaba de forma errónea (se interpretaba como normal cuando debía interpretarse como alto).

Corregido el error las conclusiones del sistema son correctas. Sin embargo, si analizamos los procesos de razonamiento empleados vemos que en la determinación del estado del $[HCO_3]^-$ no se ha tenido en cuenta el hecho de que el paciente presente o no "Fallo Renal". La presencia de fallo renal puede alterar los valores "normales" del $[HCO_3]^-$, si en nuestro estudio no aparece ningún caso con esta enfermedad el sistema parecerá funcionar perfectamente, pero sus conclusiones serán erróneas en el momento que aparezca un paciente con fallo renal (Figura 5.4).

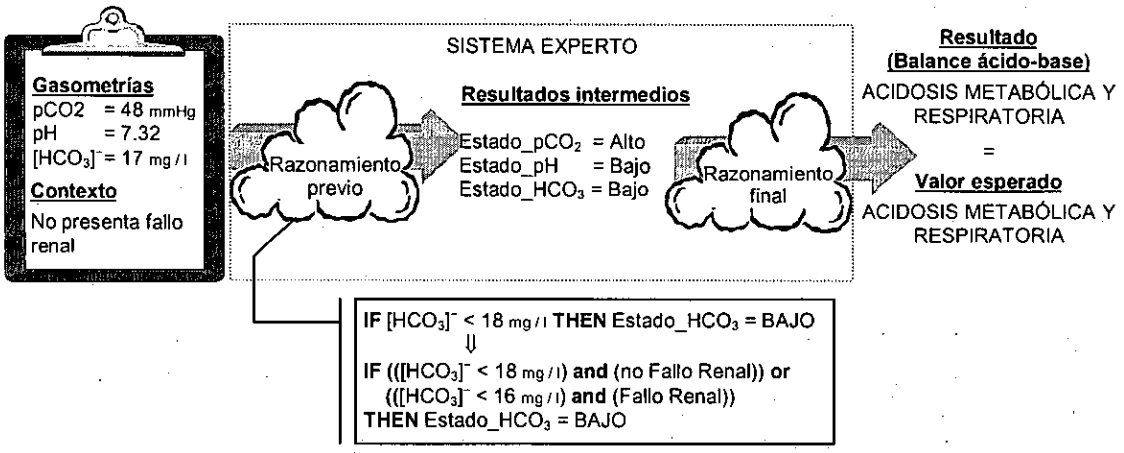


Figura 5.4. Los resultados son correctos pero en la determinación del estado del $[HCO_3]^-$ no se ha tenido en cuenta la presencia de fallo renal(para un paciente con fallo renal un $[HCO_3]^-$ de 17 mg /l sería normal y no bajo.)

Con este sencillo ejemplo hemos visto como el análisis de los resultados intermedios puede ayudar a la detección de errores en las conclusiones finales, y como una estructura de razonamiento inadecuada (en este caso incompleta) puede parecer correcta pero dar problemas cuando el ámbito de trabajo se amplía (aparecen pacientes con fallo renal).

5.3. Datos utilizados en la validación

El uso de casos de prueba es el método más ampliamente utilizado para la validación de sistemas expertos. En un mundo ideal contaríamos con una gran cantidad

de casos que representarían un rango completo de problemas, y que son analizados por una serie de expertos. En la realidad, desafortunadamente, es muy común no disponer más que de un número reducido de casos y con poco expertos que nos ayuden a analizarlos. Para que una muestra de casos sea susceptible de ser aceptada en un proceso de validación debe cumplir dos propiedades fundamentales: cantidad y representatividad.

El número de casos empleados en la validación tiene que ser suficiente para que las medidas de rendimiento que obtengamos sean estadísticamente significativas (Chandrasekaran, 1983). Ante esto podemos plantearnos un método muy sencillo de captura de datos: ir recogiendo todos los casos que podamos hasta que tengamos un número suficiente de ellos.

No obstante hay que considerar otra característica de la muestra como es su representatividad. No sólo hay que capturar un número elevado de casos, sino que éstos deber ser representativos de los problemas comunes a los que se va a enfrentar el sistema experto. Chandrasekaran (1983) aconseja que aquellos problemas que resuelva el sistema experto deben aparecer representados en los casos de prueba. O'Keefe et al. (1987) destacan que la cobertura de los casos es mucho más importante que su número, y que los casos deben representar con fiabilidad el dominio de entrada del sistema. El dominio de entrada está constituido por aquellos casos que son susceptibles de ser tratados por el sistema experto, cuanto mayor sea el dominio de entrada más compleja se hace la validación del sistema.

Para intentar mantener la representatividad de los datos se suelen emplear muestreos estratificados. Así, por ejemplo, supongamos que tenemos un sistema experto médico a partir del cual pretendemos obtener tres posibles diagnósticos: A, B y C. Revisando la historia clínica comprobamos que en este tipo de clasificaciones el diagnóstico A ha aparecido el 80 % de las veces, B el 15 % y C el 5 %. De esta forma, si nuestra muestra está compuesta por 200 casos, 160 de ellos deben pertenecer al diagnóstico A, 30 al diagnóstico B y 10 al diagnóstico C.

Aunque el procedimiento de muestras estratificadas pueda parecer válido, su utilización también ha sido objeto de controversias. Así, O'Leary (1993), presenta el ejemplo de los sistemas expertos que analizan las probabilidades de bancarrota en firmas comerciales estadounidenses. En un año se producen sólo entre un 3 y un 5 % de bancarrotas, lo que significa que en una muestra estratificada una cantidad cercana al 96% lo constituirán casos pertenecientes a firmas comerciales que no han sufrido una bancarrota. Esta *inundación* de casos puede provocar que el sistema obtenga tasas de acierto elevadas aún cuando sus capacidades para predecir una bancarrota no sean adecuadas. En tales casos puede resultar adecuado establecer una muestra equilibrada, en la que el número de casos de bancarrota sea similar al número de casos en los que no se ha producido una bancarrota.

También puede ocurrir que el experto esté interesado en comprobar la respuesta del sistema ante casos extraños y complejos que, si empleáramos una muestra estratificada, aparecerían en una proporción minúscula. En tal caso la muestra debe variarse para acoger un número representativo de casos del tipo especificado.

Otro problema que puede aparecer es que no sea posible disponer de casos de prueba para validar el sistema. Hay que recordar que en la validación siempre es

aconsejable no utilizar aquellos casos que se han utilizado en el diseño del sistema ya que, previsiblemente, el sistema habrá sido adaptado para tratar estos casos adecuadamente.

Una solución a la carencia de casos es la utilización de casos sintéticos, es decir, casos generados artificialmente por los expertos. El problema con esta aproximación es que demanda una considerable objetividad por parte de los validadores, ya que siempre es una tentación generar casos que resalten los puntos fuertes del sistema.

A pesar de todos los problemas comentados, el estudio de casos de prueba resulta adecuado para la validación de sistemas expertos porque se adapta perfectamente a los métodos de desarrollo incremental. Podemos partir de un conjunto de casos limitado para realizar la validación de las primeras etapas de desarrollo y, a medida que el sistema se vaya ampliando y se desarrollen nuevas funcionalidades, podemos capturar nuevos casos de prueba para validar las nuevas capacidades del sistema. Después de una modificación importante podemos volver a analizar casos ya resueltos para comprobar si algún *efecto lateral* ha provocado errores que, antes de la modificación, no aparecían.

5.4. Criterios de validación

La casuística empleada en la validación del sistema debe incluir dos tipos de datos: por un lado las características de cada caso en particular y, por otro lado, un criterio que permita identificar el tipo de caso que estamos tratando. Siguiendo con el ejemplo del apartado 5.2, la casuística de validación incluiría como características de cada caso los valores del pH, del pCO_2, del $[HCO_3]^-$ y la presencia o no de fallo renal. Como criterio identificativo de cada caso en particular se incluye una etiqueta que asocia cada caso con la categoría a la que pertenece (como por ejemplo "acidosis metabólica y respiratoria).

El proceso de validación se haría de la siguiente manera (Figura 5.5):

1) Se obtiene la casuística de validación.

2) Los datos de la casuística son pasados al sistema experto que se encarga de interpretarlos.

3) Los resultados del sistema y el criterio de validación que acompaña a los datos sirven de entrada para un proceso de validación en el que se analizará el rendimiento del sistema experto.

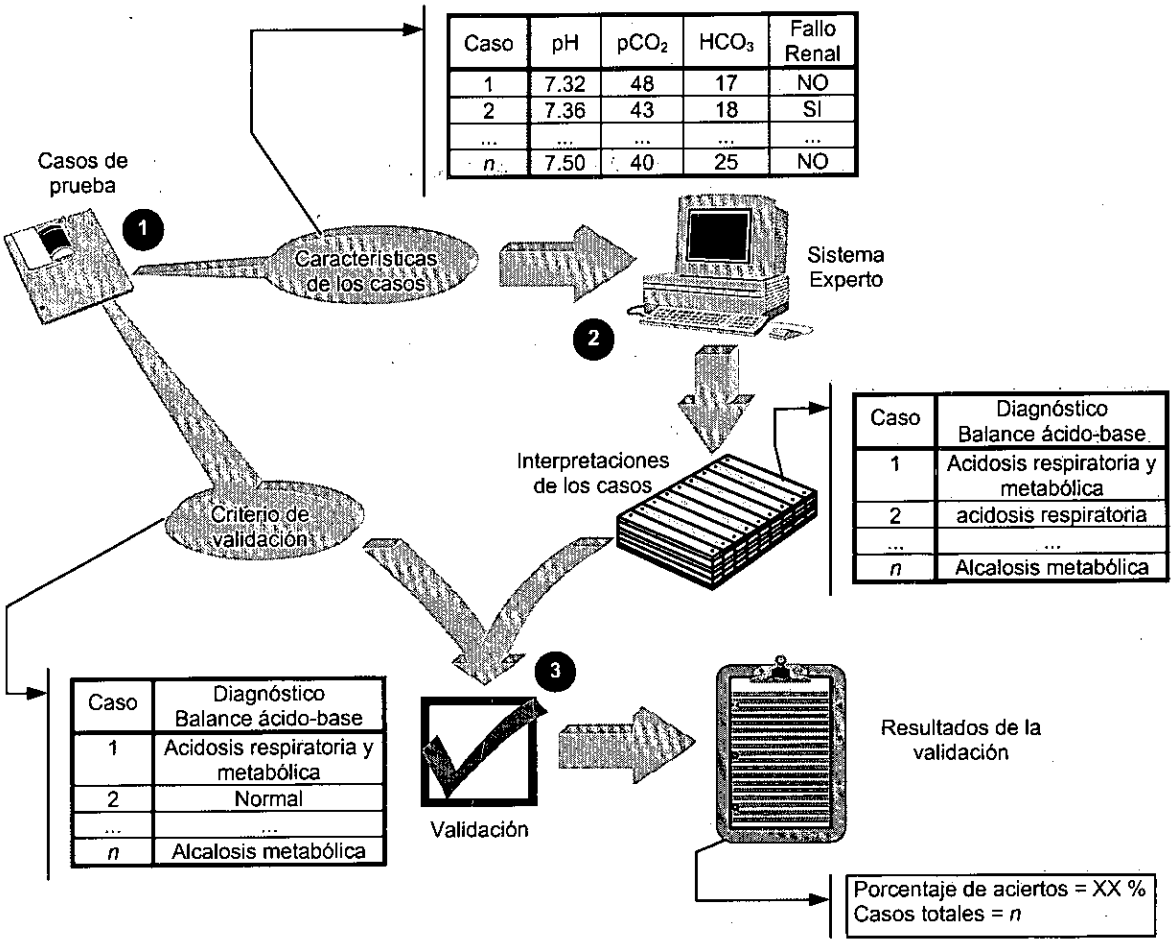


Figura 5.5. Proceso de validación a partir de casos de prueba: (1) Obtención de la casuística, (2) obtención de los resultados del sistema y (3) proceso de validación en el que se comparan los resultados del sistema con el criterio de validación..

Aunque el proceso pueda parecer sencillo existe un inconveniente importante: ¿qué utilizamos como criterio de validación?, ¿quién asocia cada caso de prueba con una categoría diagnóstica en particular?. Podemos diferenciar dos tipos de validación atendiendo al tipo de criterio establecido: validación contra el experto y validación contra el problema (Gaschnig et al., 1983).

5.4.1. Validación contra el experto

La validación contra el experto consiste, básicamente, en utilizar las opiniones y los diagnósticos de expertos humanos como criterio de validación. Este tipo de validación es la más comúnmente empleada en los sistemas inteligentes, después de todo, lo que pretendemos es construir un modelo del conocimiento del experto humano, por lo que resulta lógico utilizar a los expertos como criterio de nuestra validación.

Sin embargo, la validación contra el experto no está exenta de problemas, generalmente debidos a la propia naturaleza del conocimiento. Así, puede ser común que, expertos de un mismo nivel diagnostiquen soluciones diferentes ante el mismo problema. Incluso un mismo experto puede tener actitudes diferentes, ante un mismo caso, según las condiciones del momento en que fue realizado el análisis. Las causas de estas discordancias pueden ser:

- Factores externos: estrés, cansancio, etc.
- Los expertos pueden no ser independientes y estar influidos por otro de mayor categoría profesional, o de mayor prestigio, o de mayor poder, etc.
- Ambigüedades o errores en la adquisición de los datos pueden provocar que los expertos den opiniones diferentes ante los mismos casos.
- Los expertos pueden pertenecer a diferentes escuelas de pensamiento.
- Tendencias, a favor o en contra, de los sistemas expertos pueden hacer variar las opiniones de los expertos humanos en la validación

Existen tres posibles tipos de validación contra los expertos: (1) validación contra un experto, (2) validación contra un grupo de expertos y (3) validación contra un consenso de expertos (Figura 5.6).

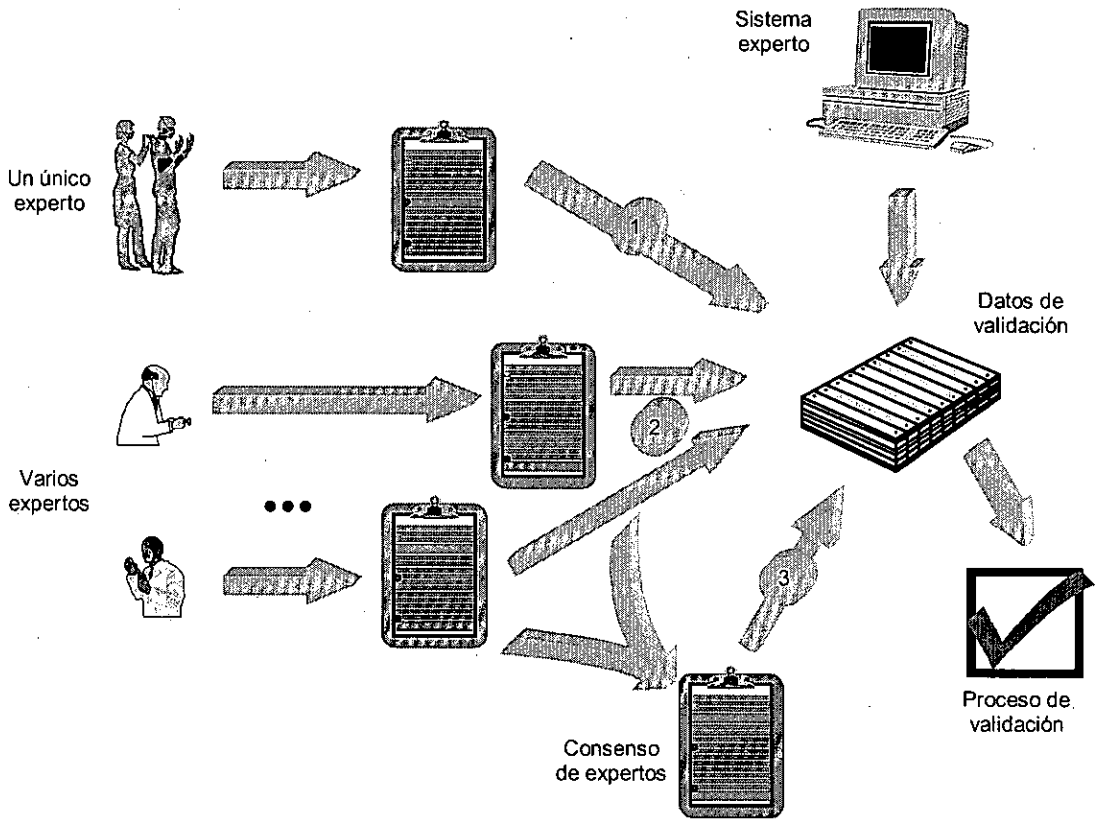


Figura 5.6. Posibles tipos de validación utilizando a los expertos como criterio: (1) con las opiniones de un único experto, (2) con las opiniones de varios expertos y (3) con las opiniones de varios expertos resumidas en un consenso..

Validación contra un único experto

La validación contra un único experto no es la más recomendable de todas pero, desgraciadamente, suele ser bastante común. Dada la escasa disponibilidad de expertos humanos, no siempre es posible contar con varios expertos en el proceso de validación. El inconveniente de utilizar un único experto es que la objetividad del estudio es cuestionable.

Validación contra un grupo de expertos

Una situación más deseable a la hora de realizar la validación es contar con las opiniones de una serie de expertos humanos. Esto conlleva una serie de ventajas: (1) no estamos ligados a una única opinión, que puede ser errónea, y (2) permite comparar el grado de consistencia existente entre los expertos del dominio.

El principal inconveniente de esta técnica es cómo medir el rendimiento del sistema experto. Generalmente los expertos no suelen tener la misma cualificación y se suele buscar una concordancia elevada con aquellos expertos de mayor nivel. Sin embargo, si los expertos son todos de un nivel similar generalmente lo que se busca es comprobar que los diagnósticos del sistema se parezcan a los diagnósticos de los expertos, tanto como los diagnósticos de los expertos se parecen entre sí. Existen una serie de medidas y procedimientos estadísticos especialmente diseñados para medir estos acuerdos (medidas de Williams, análisis cluster, escalamiento multidiménsional y medidas de dispersion y tendencia) que analizaremos más adelante.

Validación contra un consenso de expertos

La otra opción comúnmente empleada en la validación con expertos, es conseguir unir las opiniones de varios expertos en una única opinión. Este consenso tiene la ventaja de que procura ser lo más objetivo posible, y si el acuerdo del sistema inteligente con el consenso es amplio, la confianza en el sistema aumentará considerablemente. El inconveniente de esta técnica es que, en cierta manera, estamos volviendo a la técnica de validación con un único experto, es decir, todo aquello que cae fuera del consenso es considerado erróneo. Sin embargo puede haber otras soluciones válidas que los expertos podrían haber elegido, pero que han cambiado para adaptarse a un estándar del cual no están plenamente convencidos (posiblemente influidos porque un experto de mayor nivel está de acuerdo con el consenso). Además, la búsqueda de un estándar o consenso entre los expertos puede ser una ardua tarea.

Entre los distintos métodos para lograr un consenso a partir de las opiniones de varios expertos destaca el método *Delphi* (Sackman, 1974). Este método se caracteriza por el anonimato y la interactuación remota de los participantes, su perfil retroalimentado y el uso de metodologías estadísticas en el análisis de los resultados, en el que se combinan generación de ideas y evaluación de opciones. Fue desarrollado por la compañía RAND como un método de prospección, y se basa en la recopilación de información cualitativa basada en juicios de expertos. El modelo aspira a eliminar los efectos indeseables de la interactuación directa eliminando el contacto personal entre los miembros del proyecto que, ni tan siquiera, conocen la identidad de los demás miembros del grupo.

Delphi utiliza un grupo o panel de expertos, seleccionados de acuerdo con su valía profesional y la naturaleza del problema, al que se envía un cuestionario completo para recopilar juicios acerca de procesos o fenómenos reales, más específicamente sobre su tendencia y desarrollo futuro. Todos los participantes formulan de manera secreta e independiente las hipótesis e ideas que les sugiere el problema, que son enviadas por escrito al coordinador del proyecto. Éste interpreta y consolida los resultados preliminares - hipótesis, áreas de interés en relación al problema, propuestas - en un resumen global y anónimo de carácter estadístico y frecuentemente tabular.

Dicho resumen global se envía, junto con información estadística, a los expertos, a quienes se solicita que, en su caso, modifiquen sus apreciaciones iniciales o realicen las propuestas o aclaraciones que consideren oportunas, teniendo en cuenta las razones o consideraciones expuestas por los demás participantes. En la medida en que sus apreciaciones difieran de la opinión predominante del grupo se solicita que aclare su posición, lo que permite incorporar nuevas ideas y perspectivas al grupo de decisión. Tabulada la información, el coordinador envía un nuevo informe a los participantes y se reinicia el ciclo de consulta; a medida que éste se repite las posiciones de los expertos tienden a converger en las variables y procesos críticos del fenómeno, hasta alcanzar un grado suficientemente amplio de consenso. El proceso del método Delphi se representa en la Figura 5.7.

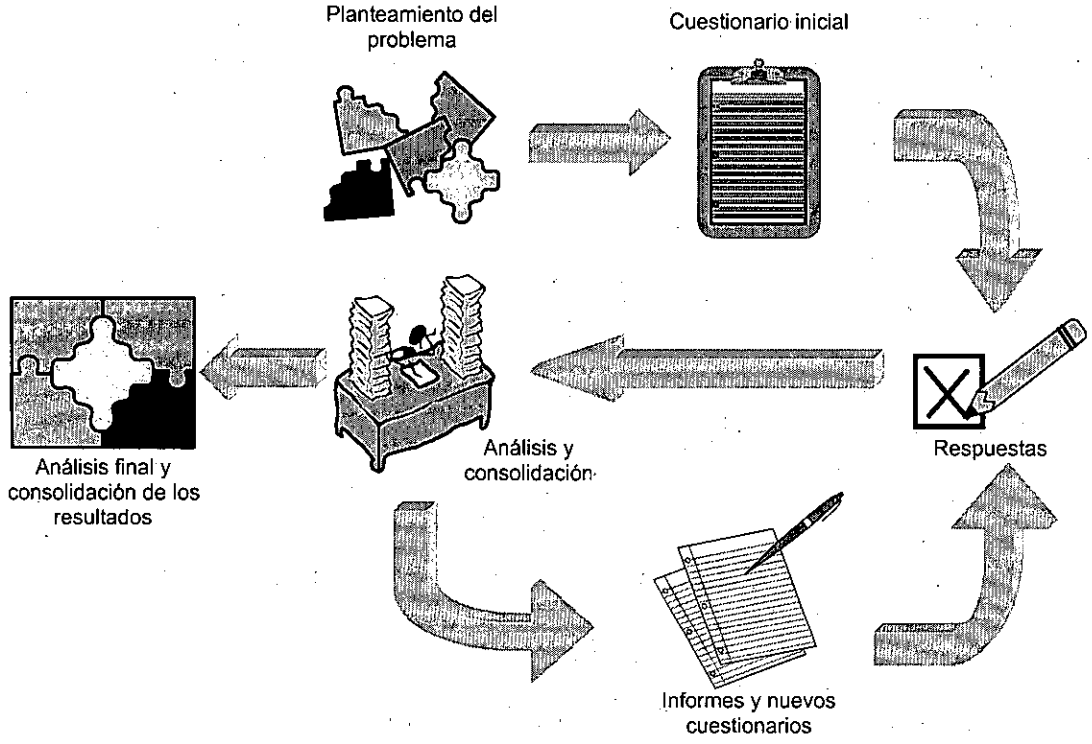


Figura 5.7. El proceso del método Delphi.

Este método se diferencia, pues, del panel de expertos tradicional en su carácter anónimo y en la realización de dos o más ciclos iterativos de consulta, lo que le confiere gran potencia en cuanto a la generación de ideas y la orientación al compromiso lograda por la información estadística que se proporciona, al término de cada ciclo, a los expertos. Entre sus deficiencias más significativas cabe destacar el elevado grado de dependencia en relación al contenido, y la expresión de las preguntas en el cuestionario inicial y la selección de los expertos.

Existen otros muchos métodos para lograr un consenso entre varios expertos como el Brainstorming, las técnicas de grupos nominales, el AHP, ... (Medsker et al., 1994). Tambien autores como (Xu et al., 1992) han propuesto métodos matemáticos para combinar información proveniente de distintas fuentes. Sin embargo el método más popular dentro de los sistemas expertos es el método Delphi debido a su carácter anónimo y a su método de realimentación controlada (Clarke et al., 1994). Podemos encontrar ejemplos de la utilización del método Delphi en sistemas expertos en (Hamilton y Breslawski, 1996) y (Roth y Wood, 1993).

5.4.2. Validación contra el problema

Leyendo el apartado anterior, una pregunta que, con toda probabilidad ha surgido es: ¿qué pasa si los expertos humanos se equivocan?. Así, si puede verse como natural que dos expertos discrepen, también puede suceder que ninguno de ellos haya sabido dar con la solución real del problema.

Supongamos el ejemplo del sistema MYCIN (Shortliffe, 1976). MYCIN se encargaba de identificar la bacteria causante de la infección de un paciente y sugerir la terapia apropiada a cada caso. Evidente MYCIN puede evaluarse comparando sus diagnósticos con los de los expertos humanos. Pero también podemos comparar los diagnósticos de MYCIN con los resultados del laboratorio que nos identifican, de forma inequívoca, cual ha sido la bacteria causante de la infección.

Este segundo tipo de validación descrito se denomina *validación contra el problema*, ya que estamos tratando de descubrir si nuestro sistema resuelve realmente el problema que le han planteado.

La ventaja de este método de validación es evidente: se trata de un método completamente objetivo, la solución real del problema es la que se muestra. Si nuestro sistema discrepa del experto pero coincide con la solución real, la credibilidad del sistema experto se verá aumentada.

Sin embargo este método también presenta inconvenientes. Uno de ellos es que podemos volver a caer en la "falacia del superhombre" que habíamos descrito con anterioridad, es decir, exigirle más al sistema experto de lo que se le exigiría al un experto humano. Así, supongamos un sistema que presenta un acuerdo del 70% con la solución real del problema. Este resultado puede parecer inadecuado, sin embargo, cuando analizamos los resultados de los distintos expertos humanos vemos que el acuerdo de estos con la solución real tampoco sobrepasa el 70% y que sus diagnósticos son muy similares a los del sistema experto. En tal caso podremos suponer que el comportamiento del sistema experto es aceptable (tal y como se había descrito en el apartado 3.4.4 al describir la metodología espiral de desarrollo de sistemas expertos).

Otro problema que surge en la validación contra el problema es que, puede no ser posible obtener una solución real. Siguiendo con el ejemplo de MYCIN, el sistema experto aconsejaba una terapia para cada caso, la única forma de comprobar que la terapia es adecuada es probarla sobre el paciente. Evidentemente, por razones éticas, solo se podrá probar una terapia sobre un paciente si coincide con la terapia que ha prescrito el experto humano (lo que limita bastante el estudio). Además el hecho de que el paciente evolucione bien puede no ser indicativo de que la terapia aplicada es la mejor (puede existir otra que haga evolucionar al paciente más deprisa y con menos sufrimiento). Por todos estos motivos la validación de MYCIN se realizó contra los expertos y no contra el problema (Yu et al., 1979).

O'Keefe et al. (1987) también recomiendan la validación contra expertos humanos, aunque indica que, si está disponible la solución real del problema, su utilización dentro del proceso de validación puede proporcionar información muy interesante.

5.5. Momento en el que se realiza la validación

Otro problema que surge a la hora de plantear la validación es: ¿cuándo realizarla?. Ante esto podemos encontrarnos dos posiciones: por un lado Bachant y McDermott (1984) advierten que validar un sistema que no está completo puede no ser útil, ya que éste no posee todo el conocimiento necesario para establecer decisiones correctas. Por otro lado Buchanan y Shortliffe (1984) recomiendan realizar la validación a lo largo de todo el desarrollo del sistema.

Como hemos visto al describir las distintas metodologías de la ingeniería del conocimiento, el punto de vista más comúnmente aceptado es el de realizar la validación a lo largo del desarrollo del sistema, realizando preferentemente un desarrollo incremental en el cual, al final de cada incremento, se realiza una validación. Sin embargo el razonamiento de Bachant y McDermott también es, en cierto sentido, válido. Así, en las primeras etapas de desarrollo, puede ser normal que el rendimiento del sistema no sea elevado y lo que se espera es que este rendimiento se eleve a medida que se va desarrollando el proyecto.

La validación que se realiza en etapas tempranas del desarrollo esta muy vinculada al proceso de adquisición del conocimiento. Así surge un nuevo proceso, denominado *refinamiento del conocimiento*, y que podemos encuadrar dentro de la fase de adquisición. El proceso de refinamiento consiste en verificar y validar el conocimiento recién adquirido en busca de problemas, resultados incorrectos, estructuras inadecuadas, etc. Existen herramientas como SEEK (Politakis, 1985) y SEEK2 (Ginsber y Weiss, 1985) que se encargan de verificar y validar el nuevo conocimiento adquirido identificando las reglas que pueden ser causa de los errores.

Otro aspecto a tener en cuenta, y que guarda cierta relación con el momento de realizar la validación, es la diferenciación existente entre la llamada validación retrospectiva y la validación prospectiva. La *validación retrospectiva* se realiza sobre casos históricos ya resueltos y almacenados en una base de datos. Este tipo de validación es la más comúnmente realizada en los sistemas expertos y los casos utilizados pueden incluir como referencia de validación tanto opiniones de expertos humanos, como la solución real al problema planteado (validación contra expertos o contra el problema). La validación retrospectiva se utiliza en las etapas de desarrollo del sistema, antes de que este se instale en su campo de trabajo habitual.

Por otro lado, la *validación prospectiva* consiste en confrontar al sistema con casos reales y ver si es capaz de resolverlos o no (está frecuentemente relacionada con la validación orientada al problema). En la validación prospectiva no se utilizan casos almacenados en bases de datos sino que se utilizan casos que en ese momento están siendo tratados por expertos humanos. De este modo se puede evaluar, no sólo la corrección de los resultados, sino aspectos referentes al uso del sistema. El problema surge, al igual que en la validación contra el problema, cuando el dominio de aplicación es crítico y el sistema intenta manipular el entorno (por ejemplo, administrándole una terapia a un paciente). Si la manipulación no ha sido aprobada por un experto humano no podrá llevarse a cabo, lo que puede limitar bastante este tipo de validación.

La validación prospectiva se utiliza una vez que hemos validado el sistema en un entorno de desarrollo y utilizando casos históricos, y se desea realizar una nueva

validación en el campo de aplicación del sistema. Este tipo de validación es similar a las pruebas beta que analizaremos más adelante.

5.6. Métodos de validación

Los métodos para realizar la validación pueden dividirse en dos grupos principales: métodos cualitativos y métodos cuantitativos (O'Keefe et al., 1987). Los métodos cualitativos emplean técnicas subjetivas de comparación del rendimiento mientras que los métodos cuantitativos se basan en la utilización de medidas estadísticas. Esto no implica que los métodos cualitativos sean menos formales que los métodos cuantitativos. Estas técnicas no son mutuamente excluyentes y lo normal es utilizar una combinación de las mismas.

5.6.1. Métodos cualitativos

Dentro de los métodos cualitativos de validación podemos destacar los siguientes:

Validación superficial.

Es una validación informal que se basa en reuniones entre los desarrolladores del sistema experto, algún experto humano del dominio y, ocasionalmente, algún usuario. En dichas reuniones se analizan, subjetivamente, las conclusiones a las que llega el sistema cuando es confrontado con una serie de casos de prueba.

Este tipo de validación es comúnmente empleado en la fases de desarrollo del sistema experto pero se recomienda que no sea el único tipo de validación aplicada al sistema.

Pruebas de Turing

Uno de los principales problemas de la colaboración de expertos en la validación de sistemas inteligentes es la presencia de tendencias a favor o en contra del propio sistema, o de las opiniones de otros expertos. Con objeto de evitar estas tendencias se utilizan las pruebas de Turing, desarrolladas a partir de la idea propuesta por el matemático Alan Turing en 1950.

En estas pruebas lo que se pretende es reunir a un grupo de expertos humanos y hacer que estos analicen los resultados de sus compañeros y los resultados del sistema experto. Estos resultados son presentados de tal forma que resulta imposible conocer la identidad de la persona, o máquina, que los ha realizado.

Chandrasekaran (1983) recomienda la utilización de estas pruebas en sistemas expertos médicos y muestra una metodología para llevarlas a cabo, que incluye la validación de los resultados finales del sistema así como sus estructuras de razonamiento. Las pruebas de Turing también se han empleado en la validación de los sistemas expertos MYCIN (Yu et al., 1979) y ONCOCIN (Hickam et al., 1985).

Test de campo

Los tests de campo consisten en colocar al sistema experto en el que va a ser su entorno de trabajo habitual y permitir que los usuarios interaccionen con él en busca de posibles errores. Es lo que en la ingeniería del software conocíamos como pruebas beta.

Los tests de campo presentan una serie de ventajas como: (1) parte de las tareas de validación se efectúan por los usuarios del sistema, (2) el nivel de rendimiento aceptable se obtiene implícitamente (cuando los usuarios dejan de notificar problemas) y, (3) permite descubrir errores que se habían pasado por alto en otro tipo de validaciones.

Sin embargo su utilización conlleva una serie de problemas: (1) los usuarios pueden inundarnos con llamadas sobre preguntas menores que tienen poca relación con el rendimiento del sistema en si, (2) el sistema puede perder credibilidad si el prototipo mostrado es muy incompleto, y (3) sólo puede utilizarse en aquellos dominios no críticos en los que los usuarios están capacitados para comprobar la corrección de las conclusiones del sistema experto.

Un ejemplo de la utilización de los tests de campo puede verse en la validación del sistema R1/Xcon (Bachant y McDermott, 1984)

Validación de subsistemas

Este método requiere la división de la base de conocimientos en diversos subsistemas o módulos que, posteriormente, se validan por separado utilizando otros métodos.

Esta técnica de "divide y vencerás" permite reconocer más fácilmente los errores y facilita el proceso de validación. Sin embargo, presenta una serie de inconvenientes como son: (1) no todos los sistemas se pueden dividir fácilmente en subsistemas independientes y, (2) la validación de todos los subsistemas por separado no es equivalente a la validación del sistema completo. Por ejemplo, supongamos dos módulos de un sistema experto médico que diagnóstican por separado la administración de dos drogas distintas. La administración de las drogas por separado no ofrece problemas, sin embargo, su administración conjunta puede ser peligrosa para la vida del paciente.

Análisis de sensibilidad

Esta técnica consiste en presentar, a la entrada del sistema, una serie de casos muy similares entre sí, conteniendo sólo pequeñas diferencias. El impacto de dichas variaciones en los casos de entrada puede ser estudiado observando los cambios resultantes en la salida.

Esta técnica es especialmente útil cuando tratamos sistemas que manejan medidas de incertidumbre, ya que puede estudiarse el impacto de los cambios en dichas medidas, tanto en los resultados intermedios, como en las conclusiones finales.

Grupos de control

Los sistemas expertos pretenden simplificar el trabajo a realizar por parte de los expertos humanos. Por ello, no sólo debe evaluarse el sistema por separado, también es útil comprobar el impacto que tiene el sistema en la organización. Una técnica comúnmente empleada en este cometido es la basada en grupos de control.

En este caso se presentan los casos a dos grupos de expertos, unos que utilizan el sistema experto y otros que trabajan sin él (y constituyen el denominado grupo de control). De esta forma podemos comparar el rendimiento de los expertos cuando utilizan el sistema experto, y cuando no lo utilizan. Para una mayor discusión sobre los grupos de control y otras técnicas denominadas *quasi-experimentales* se puede consultar (Adelman, 1991b)

5.6.2. Métodos cuantitativos

La validación cuantitativa consiste en el empleo de medidas estadísticas para cuantificar el rendimiento de un sistema experto. Muchos métodos estadísticos se han empleado en la validación: contrastes de hipótesis, análisis de la varianza (ANOVA), intervalos de confianza, etc. Sin embargo, estas técnicas pueden resultar confusas para alguien que no tenga amplios conocimientos en la estadística, y son difíciles de interpretar.

En nuestro estudio, hemos decidido centrarnos en aquellas técnicas más comunes y fáciles de interpretar (como el porcentaje de acuerdo), que utilizadas junto a otras medidas y técnicas gráficas permiten tener un conocimiento amplio sobre el rendimiento de nuestro sistema.

Podemos dividir nuestras técnicas cuantitativas en tres grupos: medidas de pares, medidas de grupo y ratios de acuerdo. A continuación haremos una breve descripción de todas ellas, y se analizarán con más detalle en el siguiente capítulo. Aparte de estas técnicas, en este apartado también comentamos otras comúnmente usadas pero no incluidas en la metodología que proponemos en el capítulo 7, como principal aportación de esta tesis.

Medidas de pares

Las medidas de pares pretenden evaluar el grado de acuerdo y/o asociación entre los diagnósticos de dos expertos (incluyendo sistema experto, expertos humanos, o una referencia estándar).

Las medidas de acuerdo se consideran un tipo especial de medidas de asociación destinadas a comprobar la fiabilidad o la reproductibilidad de las observaciones realizadas. Dentro de ellas destacamos: el porcentaje de acuerdo, el porcentaje de acuerdo dentro de uno (similar al anterior pero que considera acuerdos parciales aquellas discrepancias que se diferencias en una sólo categoría semántica), kappa (que corrige aquellos acuerdos que son debidos a la casualidad) y kappa ponderada (similar a la anterior pero ponderando las distintas discrepancias en función de la magnitud de las desviaciones).

En las medidas de asociación se investiga el grado de relación lineal existente entre las variables y, si es posible, predecir los valores de una variable a partir de los valores de otra. Dentro de estas medidas podemos destacar la tau (τ) y la tau b (τ_b) de Kendall, la rho de Spearman (r_s) y la gamma de Goodman-Kruskal (γ). Todas ellas medidas no paramétricas, ya que no hacen ninguna suposición sobre la distribución subyacente de los datos.

Medidas de grupo

Las medidas de grupo tratan la información de un grupo de expertos sobre una determinada interpretación. El objetivo es ver qué grupos pueden aparecer dentro de los expertos y comprobar si las opiniones del sistema experto son similares a las de los otros expertos (especialmente de aquellos que se consideran con mayor categoría). Dentro de las medidas de grupo podemos destacar:

- *Las medidas de Williams*, Con las que se pretende ver si la relación de un experto aislado con el grupo de referencia es similar a la relación existente entre los expertos del grupo.

- *El análisis cluster*, en donde agrupamos a los expertos en un árbol jerárquico según la similitud de sus interpretaciones.

- *El escalamiento multidimensional*, donde se representa a los expertos en un plano 2D, en el que cuanto más cercano estén los expertos más similares serán sus conclusiones.

- *Las medidas de dispersión y tendencias*, en donde se analiza cómo de dispersos son los resultados de un experto si los comparamos con el resto de expertos del grupo, y hacia qué interpretaciones tienden a dirigir sus conclusiones los expertos.

Ratios de acuerdo

Los ratios de acuerdo se encargan de comparar las interpretaciones de un experto (o sistema experto) con una referencia estándar (ya sea un consenso entre los expertos humanos o la solución real al problema planteado). Esta comparación se hace para cada una de las posibles categorías en las que se divide una interpretación (por ejemplo, si el diagnóstico de una enfermedad puede obtener 3 posibles valores: enfermedad A, B o C, podemos hallar los ratios de acuerdo para cada una de las 3 posibles enfermedades).

Existen cuatro ratios de acuerdo (verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos) que se calculan a partir de una tabla 2×2. De esta tabla también podemos obtener otros índices como el porcentaje de acuerdo o la medida de Jaccard.

Existen muchas más medidas cuantitativas que no se han considerado adecuado incluir en la metodología que describimos en el capítulo 7, sin embargo, queremos destacar algunas de ellas por haber sido utilizadas en distintas validaciones de sistemas expertos. Estas medidas son los coeficientes de exactitud, las curvas ROC y las distancias artiméticas.

Coeficientes de exactitud

En los sistemas expertos puede ser común que una salida se represente mediante una etiqueta semántica y una probabilidad (o un factor de certidumbre) asociada. Por ejemplo, imaginemos que un sistema experto predice que un paciente tiene la enfermedad X con una probabilidad del 0.75, y otro sistema experto también predice que el paciente tiene la enfermedad X pero esta vez con probabilidad 0.95. Si el paciente efectivamente tenía la enfermedad X ambas respuestas pueden considerase correctas, pero la respuesta del segundo sistema experto será más correcta que la del primero.

Para cuantificar estas diferencias se ha propuesto la utilización de coeficientes de exactitud (Shapiro, 1977). Reggia (1985) utilizó el coeficiente Q para la validación del sistema experto TIA (un sistema para la detección de ataques isquémicos transitorios). Este coeficiente se define como:

$$Q = \frac{2\sum_{i=1}^{n}(p_i - 0.5)}{n}$$

equ. 5.1

en donde p_i es la probabilidad asignada por el sistema experto a la salida i-ésima. La medida Q se interpreta de la siguiente forma: su valor es 1 si predicción es perfecta, 0 si el sistema no presenta capacidades de predicción y −1 si la predicción es totalmente incorrecta.

Curvas ROC

Las curvas ROC (Receiver Operating Characteristic) están muy relacionadas con los ratios de acuerdo y el análisis de sensibilidad. Se utilizan, sobre todo, para analizar cómo un determinado criterio de decisión interno afecta al rendimiento del sistema. Para descubrir esta influencia se computan los ratios de acuerdo para varias situaciones en las que se ha variado el criterio de decisión interno sobre su posible rango. El gráfico que relaciona los verdaderos positivos con los falsos positivos es lo que se conoce como curva ROC.

Como ejemplo tomemos el gráfico de la Figura 5.8. En él, un umbral interno del sistema se varía desde el valor 0.9 hasta el valor 0.05. La mejor relación entre los verdaderos positivos (TP) y los falsos positivos (FP) se da cuando el valor del umbral es 0.1.

Ejemplos de la utilización de curvas ROC en sistemas reales pueden encontrarse en (Adlassnig y Scheithauer, 1989) y (Detrano et al., 1992)

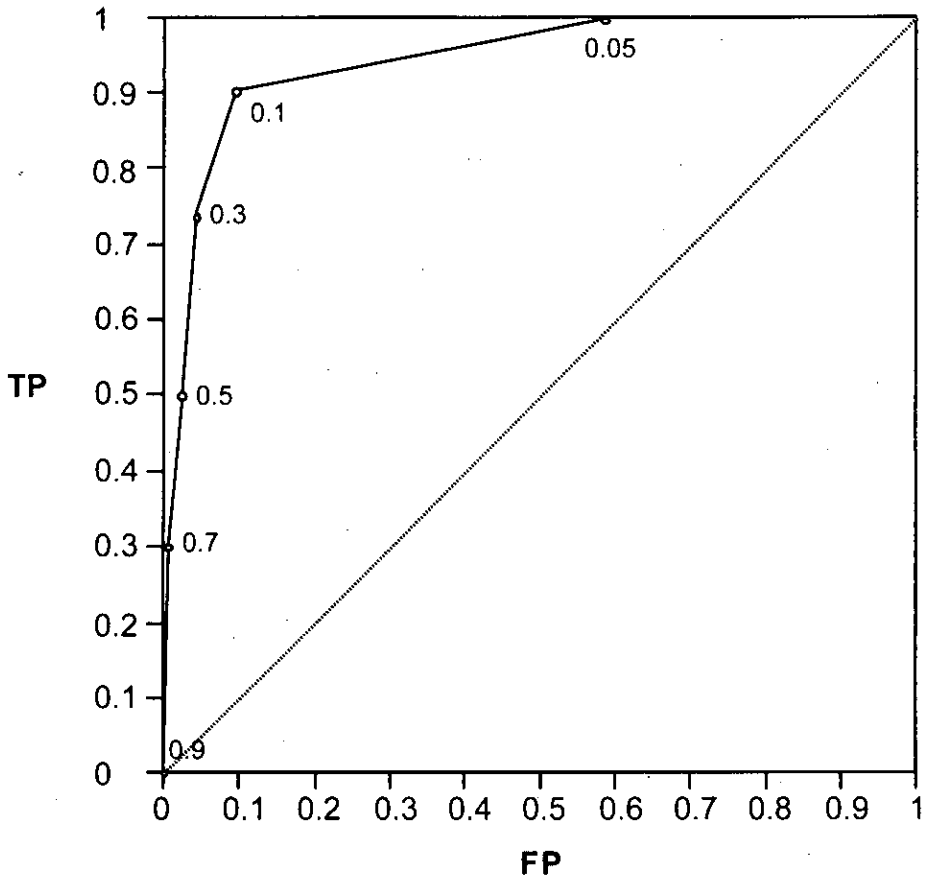


Figura 5.8. Ejemplo de curva ROC.

Distancias aritméticas

Existen ocasiones en las que se pueden medir las diferencias entre dos expertos a partir de distancias aritméticas (como la distancia euclídea). Un ejemplo de utilización de esta medida lo encontramos en la validación de los sistemas expertos probabilísticos PNEUMON-IA (Verdaguer et al., 1992) y RENOIR (Hernández et al., 1994). En ambos trabajos la salida del sistema experto es un vector en el que se detallan las probabilidades de aparición de las distintas hipótesis tomadas en consideración. Para comparar las salida del sistema experto con la de varios expertos humanos se utiliza la distancia euclídea, la distancia de Mahalanobis, la distancia de Chebychev o la distancia Manhattan (o de bloques de casas). La definición de estas medidas es la siguiente:

- Euclídea

$$d(i,j) = \sqrt{\sum_{m=1}^{N} (x_{im} - x_{jm})^2}$$

equ. 5.2

- Mahalanobis

$$d(i,j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)' \times \mathbf{W}^{-1} \times (\mathbf{x}_i - \mathbf{x}_j)}$$

equ. 5.3

- Chebychev

$$d(i,j) = \mathrm{Max}_m \left| x_{im} - x_{jm} \right|$$

equ. 5.4

- Manhattan (o bloques de casas)

$$d(i,j) = \sum_{m=1}^{N} x_{im} - x_{jm}$$

equ. 5.5

en donde d representa la distancia, i y j son expertos, N es el número de coordenadas (en este caso de posibles hipótesis) y x_{im} es la probabilidad que ha asignado el experto i a la hipótesis m. Para el caso de la distancia de Mahalanobis $(\mathbf{x}_i - \mathbf{x}_j)$ es el vector columna m-dimensional de diferencias entre los vectores de probabilidades del experto i y del experto j, $(\mathbf{x}_i - \mathbf{x}_j)'$ es el vector transpuesto correspondiente y $\mathbf{W}^{-1}$ es la inversa de la matriz de varianzas y covarianzas para las distintas enfermedades. En (Bisquerra, 1989) o en (Cox y Cox, 1994) podemos encontrar otros tipos de distancias matemáticas.

5.7. Tipos de errores en la validación

Los sistemas expertos pueden cometer dos tipos de errores: errores de comisión y errores de omisión (Gonzalez y Dankel, 1993).

Los *errores de comisión* ocurren cuando el sistema experto deduce conclusiones incorrectas a partir de los datos de entrada. Estos errores afectan al la precisión del sistema y son fáciles de detectar pero, a menudo, difíciles de localizar y corregir.

Los *errores de omisión* ocurren cuando el sistema es incapaz de llegar a ninguna conclusión a partir de los datos de entrada. En otras palabras, el conocimiento necesario para resolver un problema en particular dentro del dominio de aplicación no se encuentra en la base de conocimientos. Estos errores son más difíciles de detectar porque el caso de prueba necesario para detectarlo puede no ser evidente para el desarrollador. Estos errores afectan a la adecuación del sistema.

Sin embargo los errores también pueden aparecer en el proceso de validación. O'Keefe et al. (1987) identifican dos posibles errores: de Tipo I o de riesgo para el desarrollador y de Tipo II o de riesgo para el usuario (Tabla 5.1).

		Estado del sistema experto	
		El sistema es válido	El sistema NO es válido
Acción	El sistema se acepta como válido	Decisión correcta	Error Tipo II (riesgo para el usuario)
	El sistema NO se acepta como válido	Error Tipo I (Riesgo para el desarrollador)	Decisión correcta

Tabla 5.1. Posibles errores en el proceso de validación.

Los errores de Tipo I se producen cuando un sistema es considerado como inválido, aun a pesar de ser válido. Este error aumenta innecesariamente los costes de desarrollo del sistema y merma la credibilidad en el mismo. Se denominan como de

"riesgo para el desarrollador" porque el propio desarrollo del sistema puede ponerse en entredicho.

Por otro lado, los errores de Tipo II se producen cuando se acepta como válido un sistema que no lo es. Las consecuencias de este error son más peligrosas que las del error de Tipo I, sobre todo si el sistema actúa en dominios críticos (un sistema experto médico que diagnostique una enfermedad incorrectamente puede provocar a los pacientes un sufrimiento innecesario).

5.8. Resumen

Este capítulo constituye una introducción a las principales características de la validación de los sistemas expertos (centrándonos en una validación orientada a los resultados). Entre los distintos aspectos que caracterizan la validación destacamos los siguientes:

- *personal involucrado*, ingeniero del conocimiento, expertos humanos, evaluadores independientes o usuarios finales.
- *partes del sistema a validar*, resultados finales, resultados intermedios o los mecanismos de razonamiento.
- *datos utilizados*, muestras aleatorias, muestras estratificadas, etc.
- *criterios de validación*, validación contra el experto o validación contra el problema.
- *momento en que realizar la validación*, al final del desarrollo, durante el desarrollo.
- *métodos utilizados*, métodos cualitativos o métodos cuantitativos.
- *errores cometidos*, errores de comisión u omisión, riesgo para el desarrollador o riesgo para el usuario.

Todas estas características permiten hacer una idea global sobre la problemática que conlleva el proceso de validación de los sistemas expertos. En el próximo capítulo explicaremos en detalle los métodos estadísticos utilizados en la validación cuantitativa y que posteriormente incorporaremos en la metodología propuesta.

6. UNA REVISIÓN SOBRE MÉTODOS ESTADÍSTICOS POTENCIALMENTE ÚTILES EN LA VALIDACIÓN

Los números son como la gente. Tortúralos lo suficiente y te lo dirán todo.
Anónimo

La estadística es una ciencia según la cual todas las mentiras se tornan cuadros.
Dino Segre, Pitigilli (Escritor Italiano. 1893 – 1975)

Podría probar a Dios estadísticamente.
George Gallup (Sociólogo Estadounidense. 1901– 1984)

Sólo me fío de las estadísticas que he manipulado
Winston Churchill (Político inglés. 1.874 – 1.965)

En nuestra metodología de validación de sistemas expertos intentamos siempre utilizar medidas estadísticas de uso generalizado, y fáciles de interpretar, e incluir también métodos gráficos de representación que permitan averiguar cómo se comporta el sistema que estamos validando, sin necesidad de recurrir a complejas interpretaciones estadísticas.

En este capítulo describiremos las técnicas cuantitativas introducidas en el apartado anterior y que se dividen en tres grupos: medidas de pares, medidas de grupo y ratios de acuerdo.

6.1. Medidas de pares

Las medidas de pares pretenden evaluar el grado de acuerdo y/o asociación entre los resultados de dos expertos (incluyendo sistema experto, expertos humanos o una referencia estándar).

El desarrollo de una medida de pares parte de una base de datos de validación, en la que se incluyen los resultados del análisis de una serie de casos por parte de unos expertos. Cada experto realiza una evaluación de cada caso y asigna una etiqueta semántica determinada. El conjunto de etiquetas semánticas debe ser exhaustivo (tiene que existir una para cada posible caso), y las correspondientes etiquetas ser mutuamente excluyentes (a un caso sólo podemos asignarle una única etiqueta semántica).

Una vez tenemos nuestra base de datos de validación las medidas de pares se deben obtener según el siguiente procedimiento: (1) se desarrolla una tabla o matriz de contingencia que relaciona los resultados de los expertos en consideración, y (2) se extrae de la tabla de contingencia la medida de pares determinada (Figura 6.1).

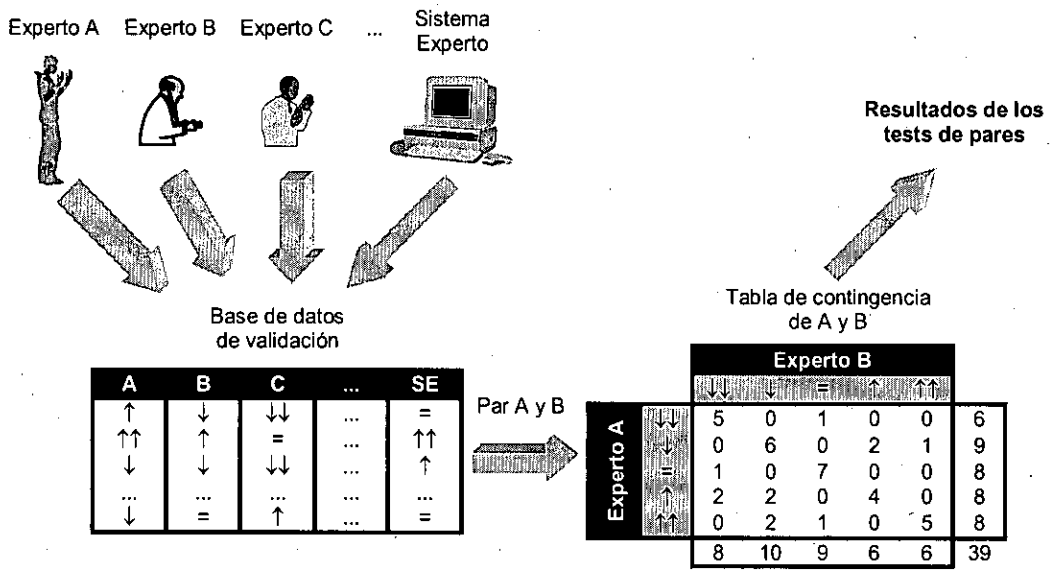


Figura 6.1 Proceso de realización de los tests de pares.

6.1.1.Tablas de contingencia

Una tabla de contingencia es una tabla que relaciona de forma cruzada datos categóricos. El origen del término *contingencia* se debe a Pearson (1904) que, originalmente, lo asignó a la medida de la desviación total de la "probabilidad de independencia" en una tabla $s \times t$. Posteriormente el término fue utilizado para referirse a la misma tabla.

En nuestro caso utilizaremos las tablas de contingencia para relacionar los resultados de dos expertos, como vemos en la Tabla 6.1.

		Resultados experto B				
		1	2	...	k	Totales
Resultados experto A	1	n_{11}	n_{12}	...	n_{1k}	$n_{1.}$
	2	n_{21}	n_{22}	...	n_{2k}	$n_{2.}$
	...	...	...	...	...	...
	k	n_{k1}	n_{k2}	...	n_{kk}	$n_{k.}$
	Totales	$n_{.1}$	$n_{.2}$	...	$n_{.k}$	$n_{..} = N$

Tabla 6.1. Tabla de contingencia que relaciona los resultados de los expertos A y B..

Esta tabla representa a dos expertos (A y B) que se encargan de asignar a cada caso una categoría semántica escogida de un conjunto de k posibles categorías. Cada celda de la tabla incluye un cantidad n_{ij} que representa el número de casos en los que el experto A selecciona la categoría i, mientras que el experto B selecciona la categoría j. Estas cantidades también se conocen con el término de frecuencias absolutas, o frecuencias absolutas observadas.

El número total de casos pertenecientes a la fila i se denota con el término $n_{i.}$ mientras que el número total de casos de la columna j se denota con el término $n_{.j}$. Estos valores se denominan frecuencias absolutas marginales (porque se suelen situar en los márgenes de la tabla), y se pueden obtener de los valores de las celdas de la siguiente forma:

$$n_{i.} = n_{i1} + n_{i2} + \ldots + n_{ik} = \sum_{j=1}^{k} n_{ij}$$

equ. 6.1

$$n_{.j} = n_{1j} + n_{2j} + \ldots + n_{kj} = \sum_{i=1}^{k} n_{ij}$$

equ. 6.2

$$n_{..} = \sum_{i=1}^{k}\sum_{j=1}^{k} n_{ij} = \sum_{i=1}^{k} n_{i.} = \sum_{j=1}^{k} n_{.j}$$

equ. 6.3

El término $n_{..}$ representa el número total de casos de la muestra y normalmente es indicado con la letra N. Esta notación se conoce como *notación de puntos*, en la que un punto en un subíndice indica que se realiza una suma sobre ese subíndice.

Supongamos que disponemos de una base de datos de validación en la que cuatro expertos humanos (A, B, C y D) y un sistema experto (SE) cuantifican el valor de una determinada variable en base a cinco posibles etiquetas semánticas (Tabla 6.2).

Casos	A	B	C	D	SE
1	ALTO	ALTO	ALTO	ALTO	ALTO
2	ALTO	BAJO	ALTO	ALTO	BAJO
3	BAJO	BAJO	NORMAL	NORMAL	BAJO
4	NORMAL	BAJO	NORMAL	NORMAL	NORMAL
5	MUY ALTO	MUY ALTO	ALTO	ALTO	MUY ALTO
6	BAJO	BAJO	BAJO	BAJO	NORMAL
7	MUY BAJO	MUY BAJO	NORMAL	BAJO	BAJO
8	NORMAL	NORMAL	NORMAL	ALTO	NORMAL
9	NORMAL	NORMAL	BAJO	MUY BAJO	NORMAL
10	BAJO	BAJO	BAJO	ALTO	BAJO

Tabla 6.2. Base de datos de validación con 4 expertos humanos (A, B, C y D) y el sistema experto (SE).

Si queremos comparar los resultados del sistemas experto con los del experto A debemos, en primer lugar, construir la tabla de contingencia que relaciona sus resultados (Tabla 6.3).

		Experto SE					
		MUY BAJO	BAJO	NORMAL	ALTO	MUY ALTO	
Experto A	MUY BAJO	0	1	0	0	0	1
	BAJO	0	2	1	0	0	3
	NORMAL	0	0	3	0	0	3
	ALTO	0	1	0	1	0	2
	MUY ALTO	0	0	0	0	1	1
		0	4	4	1	1	10

Tabla 6.3. Tabla de contingencia entre los expertos A y ES.

También es posible generar las tablas de contingencia basándonos en las frecuencias relativas, o proporciones, en vez de en las frecuencias absolutas. La frecuencia relativa de la celda *ij* (representada por p_{ij}) no es más que el número de casos de esa celda (n_{ij}) dividido por el número de casos totales (N):

$$p_{ij} = \frac{n_{ij}}{N}$$

equ. 6.4

De esta forma la Tabla 6.3 quedaría de la siguiente manera (Tabla 6.4):

		Experto SE					
		MUY BAJO	BAJO	NORMAL	ALTO	MUY ALTO	
Experto A	MUY BAJO	0.0	0.1	0.0	0.0	0.0	0.1
	BAJO	0.0	0.2	0.1	0.0	0.0	0.3
	NORMAL	0.0	0.0	0.3	0.0	0.0	0.3
	ALTO	0.0	0.1	0.0	0.1	0.0	0.2
	MUY ALTO	0.0	0.0	0.0	0.0	0.1	0.1
		0.0	0.4	0.4	0.1	0.1	1

Tabla 6.4 *Tabla de contingencia entre los expertos A y ES utilizando proporciones en vez de frecuencias absolutas.*

Las tablas de contingencia se utilizarán como base para calcular las medidas de pares. Dichas medidas pueden clasificarse en dos grupos: medidas de acuerdo y medidas de asociación. Las medidas de acuerdo se consideran un tipo especial de medidas de asociación, destinadas a comprobar la fiabilidad o la reproducibilidad de las observaciones realizadas. Por otro lado, en las medidas de asociación lo que se investiga es el grado de relación lineal existente entre las variables, y si es posible predecir los valores de una variable a partir de los valores de la otra.

6.1.2. Medidas de acuerdo

Un desacuerdo honesto es a menudo un buen signo de progreso
Gandhi (Pacifista y líder nacionalista indio. 1869-1948)

Cada vez que la gente está de acuerdo conmigo siento que debo estar equivocado
Oscar Wilde (Novelista, poeta y autor teatral de origen irlandés. 1854-1900)

Las medidas de acuerdo que estudiaremos aquí serán el porcentaje de acuerdo, el porcentaje de acuerdo dentro de uno, kappa y kappa ponderada.

6.1.2.1. Porcentaje de acuerdo

Una de las medidas de acuerdo más comúnmente utilizada es el porcentaje o la proporción de acuerdo. Ésta es simplemente el cociente entre el número de observaciones de acuerdo y el número de observaciones totales. Para obtener esta medida de la tabla de contingencia simplemente hay que sumar las frecuencias absolutas de la diagonal principal y dividirlas por el número total de casos (o simplemente sumar las frecuencias relativas o proporciones de la diagonal principal).

$$\text{Porcentaje} = \frac{\sum_{\substack{i=1,\,j=1 \\ i=j}}^{k} n_{ij}}{N} = \sum_{\substack{i=1,\,j=1 \\ i=j}}^{k} p_{ij}$$

equ. 6.5

Para los datos de la Tabla 6.3 obtendríamos el siguiente valor:

$$\text{Porcentaje} = \frac{0+2+3+1+1}{10} = \frac{7}{10} = 0.7$$

De forma similar si utilizamos las proporciones de la Tabla 6.4 obtenemos:

$$\text{Porcentaje} = 0.0 + 0.2 + 0.3 + 0.1 + 0.1 = 0.7$$

El porcentaje de acuerdo toma valores en el intervalo [0, 1] (en el que la unidad representa el acuerdo completo y el cero el desacuerdo completo), su valor no se ve afectado por variaciones en el orden de las categorías, un experto siempre presenta acuerdo perfecto consigo mismo, el acuerdo perfecto es una relación transitiva, y se trata de una medida simétrica (no importa el orden en el que escojamos a los expertos).

La principal ventaja de esta medida es la sencillez de su interpretación, que ha hecho que su uso se extendiera en distintos campos y aplicaciones. Sin embargo, presenta el inconveniente de que no tiene en cuenta los acuerdos debidos a la casualidad. Esto puede verse cuando realizamos dos clasificaciones con distintos número de categorías (por ejemplo, una con categorías "muy bajo", "bajo", "normal", "alto" y "muy alto"; y otra con categorías "muy bajo", "normal", "muy alto"). Generalmente la clasificación con menor número de categorías tendrá un porcentaje de acuerdo mayor ya que la probabilidad de realizar asignaciones iguales por casualidad es mayor cuantas menos categorías haya (Popping, 1981).

6.1.2.2. Porcentaje de acuerdo dentro de uno

Muchas veces, cuando utilizamos escalas ordinales en nuestros diagnósticos, es normal que se produzcan discrepancias que se diferencian sólo en una categoría lingüística. Por ejemplo, si nuestro diagnóstico se divide en siete categorías (Figura 6.2a) puede no ser fácil distinguir si un caso es "Algo Bajo" o simplemente "Bajo".

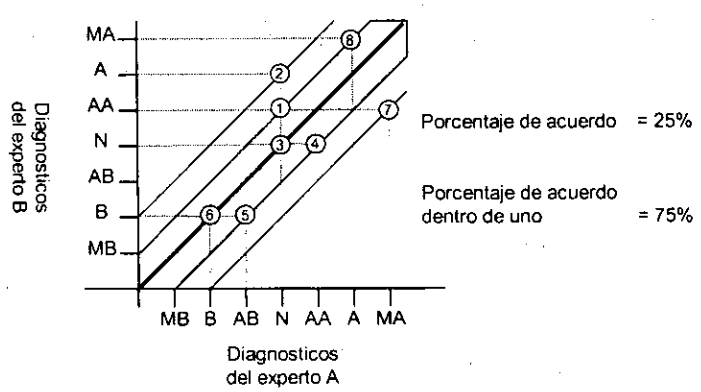
Por ello se utiliza el porcentaje de acuerdo dentro de uno, que considera como acuerdos parciales aquellos diagnósticos que se diferencian en una única etiqueta lingüística consecutiva. En la Figura 6.2b podemos ver los diagnósticos de dos expertos y en la Figura 6.2c la representación gráfica de dichos diagnósticos.

Escala semántica para la clasificación simbólica de una determinada variable.	Muy Bajo (MB)	Bajo (B)	Algo Bajo (AB)	Normal (N)	Algo Alto (AA)	Alto (A)	Muy Alto (MA)

(a)

Casos	Experto A	Experto B
1	Normal	Algo Alto
2	Normal	Alto
3	Normal	Normal
4	Algo Alto	Normal
5	Algo Bajo	Bajo
6	Bajo	Bajo
7	Muy Alto	Algo Alto
8	Alto	Muy Alto

(b)



(c)

Figura 6.2. (a) Escala semántica para la clasificación simbólica de una determinada variable. (b) Interpretaciones de dos expertos para esa variable. (c) Representación de las desviaciones de esas interpretaciones. El área sombreada representa el acuerdo dentro de uno y los números de la figura los casos de la tabla (b).

Formalmente podemos definir el porcentaje de acuerdo dentro de uno de forma similar al porcentaje de acuerdo pero, en este caso, sumando también las frecuencias pertenecientes a las diagonales adyacentes a la diagonal principal.

$$\text{Porcentaje de acuerdo dentro de uno} = \frac{\sum_{i=1,\, j=1}^{k} n_{ij}}{N}_{i=j,\, i=j\pm 1} = \sum_{\substack{i=1,\, j=1 \\ i=j,\, i=j\pm 1}}^{k} p_{ij}$$

equ. 6.6

Para nuestro ejemplo de la Tabla 6.4 el valor del porcentaje de acuerdo dentro de uno sería:

$$\begin{aligned}\text{Porcentaje de acuerdo dentro de uno} &= \text{diagonal}_{\text{principal}} + \text{diagonal}_{+1} + \text{diagonal}_{-1} = \\ &= (0.0 + 0.2 + 0.3 + 0.1 + 0.1) + \\ &\quad (0.1 + 0.1 + 0.0 + 0.0) + \\ &\quad (0.0 + 0.0 + 0.0 + 0.0) = 0.9\end{aligned}$$

El porcentaje de acuerdo dentro de uno sólo es útil cuando estamos tratando escalas ordinales y, como veremos más adelante, puede utilizarse para analizar tendencias en los resultados. Sin embargo, presenta el mismo problema que el porcentaje de acuerdo: no tiene en cuenta aquellos acuerdos que son debidos a la casualidad.

6.1.2.3. Índice kappa

Los primeros intentos de corregir acuerdos debidos a la casualidad se basaban en el test chi-cuadrado (χ^2) como, por ejemplo, el coeficiente de correlación *C* (Guilford, 1950). Sin embargo, es fácil demostrar que el uso de χ^2 (y consecuentemente de *C*) no es adecuado para la medida del acuerdo, ya que mide grados de asociación, y no grados de acuerdo. Dos expertos pueden estar asociados pero no precisamente en la dirección del acuerdo.

Cohen (1960) propuso una medida de acuerdo (denominada kappa) en la que se corregían aquellos acuerdos que eran debidos a la casualidad. Esta medida está basada en dos cantidades:

p_o = proporción de acuerdo observado,
p_c = proporción de acuerdo esperado debido a la casualidad.

De esta forma, $1 - p_c$ representa el máximo acuerdo posible una vez que se ha eliminado la casualidad y $p_o - p_c$ representa el acuerdo obtenido una vez que se ha eliminado la casualidad. Esto nos permite definir el índice kappa de la siguiente forma:

$$\kappa = \frac{p_o - p_c}{1 - p_c}$$

equ. 6.7

Acuerdos debidos a la casualidad

El término p_o es el porcentaje de acuerdo visto en el apartado 6.1.2.1, mientras que el término p_c es la suma de los productos de las proporciones marginales correspondientes a la diagonal principal y se expresa mediante la siguiente ecuación:

$$p_c = \sum_{\substack{i=1,\, j=1 \\ i=j}}^{k} p_{i.} p_{.j}$$

equ. 6.8

La equ. 6.8 se obtiene siguiendo los tres pasos que se representan en la Tabla 6.5. (utilizando los datos de los expertos A y SE de la Tabla 6.4):

1. Cálculo de las proporciones marginales mediante la suma de las proporciones correspondientes a las filas y a las columnas.

2. Cálculo de las proporciones de acuerdo debidas a la casualidad de cada celda mediante la multiplicación de las proporciones marginales correspondientes a dicha celda.

3. Cálculo del índice p_c mediante la suma de las proporciones de acuerdo debidas a la casualidad de las celdas pertenecientes a la diagonal principal.

Experto A \ Experto SE	MUY BAJO	BAJO	NORMAL	ALTO	MUY ALTO	
MUY BAJO	0.0 (0.0) ②	0.1	0.0	0.0	0.0	0.1 ①
BAJO	0.0	0.2 (0.12)	0.1	0.0	0.0	0.3
NORMAL	0.0	0.0	0.3 (0.12)	0.0	0.0	0.3
ALTO	0.0	0.1	0.0	0.1 (0.02)	0.0	0.2
MUY ALTO	0.0	0.0	0.0	0.0	0.1 (0.01)	0.1
	0.0 ①	0.4	0.4	0.1	0.1	1

p_o = 0.70
p_c = 0.27
κ = 0.589 ③

Tabla 6.5. Calculo de p_c. (1) cálculo de las proporciones marginales, (2) cálculo de las proporciones de acuerdo debidas a la casualidad para cada celda (puestas entre paréntesis) y (3) cálculo de p_c y, por consiguiente, de kappa.

La idea que subyace bajo estos cálculos es la siguiente: las proporciones marginales representan la distribución de los diagnósticos de los dos expertos cuando son tratados de forma independiente. Cuando se comparan sus diagnósticos se espera que su acuerdo sea mayor que el que cabría esperar si tomamos sus proporciones marginales como probabilidades (así la probabilidad de que escoja una determinada celda es el producto de las proporciones marginales correspondientes a dicha celda). Por ello se corrige el acuerdo observado (p_o) con el acuerdo debido a la casualidad (p_c).

Valores extremos de kappa

Si el acuerdo observado es igual al acuerdo debido a la casualidad el valor de kappa es cero. Si es acuerdo observado es mayor que el acuerdo debido a la casualidad el valor de kappa es positivo, siendo su límite máximo +1 (que corresponde al acuerdo perfecto, p_o =1). Si el acuerdo observado es menor que el debido a la casualidad el valor de kappa es negativo (nótese que el valor de kappa no está definido si p_c = 1). Sin embargo, su límite inferior no es tan sencillo de determinar como su límite superior, ya que depende de las distribuciones marginales. Si definimos a r_m como el grado de correlación existente entre las distribuciones marginales de los dos expertos, podemos obtener los siguientes valores mínimos de kappa (Tabla 6.6).

Valor de r_m	Valor mínimo de kappa
$r_m = 0$	$Min\ \kappa^- = -\frac{1}{k-1}$
$r_m < 0$ (poco común)	$Min\ \kappa^- > -\frac{1}{k-1}$
$r_m > 0$ (lo más habitual)	$Min\ \kappa^- < -\frac{1}{k-1}$

Tabla 6.6. Relación entre los valores mínimos de kappa y el coeficiente de correlación de las distribuciones marginales (r_m).

Donde, recordemos, k representa el número de categorías en las que se divide la interpretación tomada en consideración.

Las complejidades en la determinación del límite inferior de kappa son de interés meramente académico, ya que, al ser utilizado como índice de acuerdo entre dos expertos, las situaciones más normales son aquellas en las que el valor de kappa es positivo. Es más, un valor de kappa negativo indicaría que el error no es debido a la casualidad (por ejemplo, puede producirse un error sistemático si los expertos interpretan de distinta forma los límites de las categorías).

Un aspecto que también ha sido objeto de estudio es el límite superior de kappa. Este límite es +1 cuando el acuerdo es perfecto, es decir, cuando las proporciones de las celdas que no pertenecen a la diagonal principal son cero. Esto implica que las distribuciones marginales deben ser idénticas. Sin embargo, puede ser de interés determinar el máximo valor de kappa que se podría obtener con unas distribuciones marginales dadas. Este valor se define de la siguiente forma:

$$\kappa_M = \frac{p_{oM} - p_c}{1 - p_c} \qquad \textit{equ. 6.9}$$

en donde p_{oM} se calcula emparejando los valores marginales de cada experto, eligiendo el menor valor de cada par y después sumando los k valores resultantes. Para los datos de la Tabla 6.5 $p_{oM} = 0.0 + 0.3 + 0.3 + 0.1 + 0.1 = 0.8 \Rightarrow \kappa_M = 0.726$.

De esta forma κ_M es el máximo valor de κ permitido por los marginales y $1 - \kappa_M$ representa la cantidad de acuerdo (excluida la casualidad) que no puede conseguirse debido a las diferentes distribuciones marginales. Esta cantidad puede servir al

investigador como un indicador de lo difusos que resultan los límites entre las distintas categorías. Cohen cita el ejemplo de dos expertos con distinta formación que se decantan con más facilidad por determinadas categorías. En tal caso resulta de utilidad calcular el ratio κ / κ_M. En muchas aplicaciones, sin embargo, la cuestión de cómo afectan las distribuciones marginales al valor de kappa es de escasa relevancia ya que el desacuerdo debido a las distribuciones marginales tiene las mismas consecuencias que el desacuerdo que no es debido a estas causas, sigue siendo un desacuerdo.

Interpretación de kappa

El índice kappa es independiente del número de observaciones realizadas y del número de categorías presentes, y no se ve influido por permutaciones en las categorías. Se trata de un índice simétrico en el que el acuerdo perfecto es una relación reflexiva (un experto acuerda perfectamente consigo mismo) y transitiva (si el acuerdo entre A y B es perfecto, y también lo es el de B y C entonces también lo será el de A y C). Landis y Koch (1977) han desarrollado unas reglas básicas para su interpretación que, básicamente, se resumen en la Tabla 6.7.

Kappa	Nivel de acuerdo
< 0.00	Nulo
0.00 – 0.20	Insuficiente
0.21 – 0.40	Ligero
0.41 – 0.60	Moderado
0.61 – 0.80	Substancial
0.81 – 1.00	Casi perfecto o perfecto

Tabla 6.7. Interpretación del índice kappa por Landis y Koch (1977).

Características muestrales de kappa

Una aproximación al error estándar de kappa se puede obtener mediante la siguiente ecuación:

$$\sigma_\kappa = \sqrt{\frac{p_o(1-p_0)}{N(1-p_c)^2}}$$

equ. 6.10

Con valores de N grandes, la distribución muestral de kappa se aproxima a una normal, por lo que podemos fijar intervalos de confianza, por ejemplo:

Intervalo de confianza del 95% = $\kappa \pm 1.96\sigma_\kappa$
Intervalo de confianza del 99% = $\kappa \pm 2.58\sigma_\kappa$

Para realizar tests sobre la significación de kappa, cuando la kappa poblacional es cero, se define la siguiente ecuación:

$$\sigma_{\kappa 0} = \sqrt{\frac{p_c}{N(1-p_c)}}$$

equ. 6.11

La significación se obtiene normalizando el valor de κ a una normal (0, 1), esto se consigue dividendo κ por $\sigma_{\kappa 0}$, y refiriendo el resultado a una curva normal. Es necesario señalar que los tests de significación en kappa (así como en otras medidas de

acuerdo o asociación) son, en nuestro caso, de interés meramente académico. Esto es así porque, como mínimo, siempre se presupone que existirá una relación positiva que va más allá de la casualidad entre dos expertos humanos (Cohen, 1960). Una posible línea de investigación sería referir los tests a valores de la kappa poblacional distintos de cero.

Para un estudio más amplio de las características muestrales de kappa puede consultarse (Fleiss et al., 1969).

Inconvenientes de kappa

El problema que tiene el coeficiente kappa es que trata todas las discordancias de la misma manera, de forma que todas las celdas que no pertenecen a la diagonal principal tienen la misma penalización. Es decir, la penalización por equivocarse entre las categorías "Muy Alto" y "Muy Bajo" es similar a la penalización por equivocarse entre las categorías "Muy Alto" y "Alto". Para intentar resolver este problema se desarrolla el coeficiente kappa ponderada.

6.1.2.4. Kappa ponderada

Kappa ponderada (κ_w) fue desarrollada también por Cohen (1968) y es una medida de acuerdo que corrige aquellos acuerdos debidos a la casualidad, y pondera de forma distinta los desacuerdos encontrados.

La ponderación de los distintos desacuerdos se hace a partir de una matriz de pesos en la que, para cada posible par de categorías *ij*, se define un peso v_{ij}, que cuantifica el desacuerdo existente. A las celdas pertenecientes a la diagonal principal (que representan el acuerdo perfecto) se les suele asignar el valor 0 indicando que no existe ningún desacuerdo. El mayor valor de desacuerdo v_{max} es fijado por el investigador. Para cualquier conjunto de pesos, kappa ponderada es invariable ante transformaciones multiplicativas positivas, es decir, que kappa ponderada no cambiará de valor si sus pesos se multiplican por un valor mayor que cero.

Por ejemplo, para los expertos A y SE vistos anteriormente los pesos de desacuerdo podrían ser:

		Experto SE				
		MUY BAJO	BAJO	NORMAL	ALTO	MUY ALTO
Experto A	MUY BAJO	0	1	4	9	16
	BAJO	1	0	1	4	9
	NORMAL	4	1	0	1	4
	ALTO	9	4	1	0	1
	MUY ALTO	16	9	4	1	0

Tabla 6.8. Pesos de desacuerdo para los diagnósticos de los expertos A y SE.

Para incluir los pesos en la ecuación de kappa procedemos de la siguiente manera: en primer lugar partimos de la equ. 6.7 en la que definíamos el índice kappa a partir de la proporción de acuerdo observado y la proporción de acuerdo debido a la casualidad. Esta expresión también puede realizarse en base a proporciones de desacuerdo ($q = 1 - p$) en donde la proporción de desacuerdo observado es $q_o = 1 - p_o$ y la proporción de desacuerdo debido a la casualidad es $q_c = 1 - p_c$. Sustituyendo p_o y p_c por $(1 - q_o)$ y $(1 - q_c)$ en la equ. 6.7 obtenemos:

$$\kappa = \frac{q_c - q_o}{q_c} = 1 - \frac{q_o}{q_c}$$

equ. 6.12

Esta ecuación expresa el coeficiente kappa basado en proporciones de desacuerdo. Para el cálculo de κ_w se reemplazan dichas proporciones de desacuerdo por las proporciones de desacuerdo ponderado q'_o y q'_c que se definen como:

$$q'_o = \frac{\sum_{i=1,j=1}^{k} v_{ij} p_{oij}}{v_{max}}$$

equ. 6.13

$$q'_c = \frac{\sum_{i=1,j=1}^{k} v_{ij} p_{cij}}{v_{max}}$$

equ. 6.14

en donde p_{oij} es la proporción de acuerdo observada para la casilla ij, p_{cij} es la proporción de acuerdo debido a la casualidad correspondiente a la casilla ij, v_{ij} es el peso correspondiente a la casilla ij, v_{max} es el peso máximo de la tabla y k es el número de categorías.

Una vez obtenidos estos valores podemos calcular el coeficiente kappa ponderada según la siguiente ecuación:

$$\kappa_w = 1 - \frac{q'_o}{q'_c}$$

equ. 6.15

Sustituyendo la equ. 6.13 y la equ. 6.14 en la equ. 6.15, podemos eliminar el término v_{max} obteniendo:

$$\kappa_w = 1 - \frac{\sum_{i=1,j=1}^{k} v_{ij} p_{oij}}{\sum_{i=1,j=1}^{k} v_{ij} p_{cij}}$$

equ. 6.16

Aplicando esta ecuación a la Tabla 6.4 utilizando los pesos de la Tabla 6.8 obtenemos el siguiente resultado:

$$\sum v_{ij} p_{oij} = 0\,(0) + 1\,(0.1) + 4\,(0) + 9\,(0) + 16\,(0) + 1\,(0) + 0\,(0.2) + 1\,(0.1) + \ldots + 0\,(0.1) = 0.6$$

$$\sum v_{ij} p_{cij} = 0\,(0) + 1\,(0.04) + 4\,(0.04) + 9\,(0.01) + 16\,(0.01) + 1\,(0) + 0\,(0.12) + \ldots + 0\,(0.01) = 2.18$$

$$\kappa_w = 1 - \frac{0.6}{2.18} = 1 - 0..275 = 0.725$$

Kappa ponderada y pesos de acuerdo

Kappa ponderada también puede representarse en base a pesos de acuerdo (w_{ij}) en vez de pesos de desacuerdo (v_{ij}) como se venía haciendo hasta ahora. En este caso las celdas de la diagonal principal tendrán el peso de acuerdo máximo (w_{ij}) y en el resto de las celdas éstos valores irán decreciendo hasta que no haya ningún acuerdo (que es conveniente, aunque no necesario, representar con el peso 0).

En este caso definimos la proporción de acuerdo ponderado observado (p'_o) y la proporción de acuerdo ponderado debido a la casualidad (p'_c) de la siguiente forma:

$$p'_o = \frac{\sum_{i=1,\,j=1}^{k} w_{ij}\, p_{oij}}{w_{max}}$$

equ. 6.17

$$p'_c = \frac{\sum_{i=1,\,j=1}^{k} w_{ij}\, p_{cij}}{w_{max}}$$

equ. 6.18

La ecuación que define a kappa ponderada se obtiene de la ecuación de kappa (equ. 6.7) de la siguiente forma:

$$\kappa_w = \frac{p'_0 - p'_c}{1 - p'_c}$$

equ. 6.19

Sustituyendo la equ. 6.17 y la equ. 6.18 en la equ. 6.19 obtenemos

$$\kappa_w = \frac{\sum_{i=1,\,j=1}^{k} w_{ij}\, p_{oij} - \sum_{i=1,\,j=1}^{k} w_{ij}\, p_{cij}}{w_{max} - \sum_{i=1,\,j=1}^{k} w_{ij}\, p_{cij}}$$

equ. 6.20

Igual que con los pesos de desacuerdo, kappa ponderada es invariable ante multiplicaciones de los w_{ij} por un valor mayor que cero. En cuanto a la relación de los w_{ij} con los v_{ij}, kappa ponderada permanecerá constante siempre que los dos tipos de pesos se expresen como complementarios de las proporciones de sus respectivos valores máximos:

$$\frac{w_{ij}}{w_{max}} = 1 - \frac{v_{ij}}{v_{max}}$$

lo que nos lleva a que

$$w_{ij} = \left(\frac{w_{max}}{v_{max}}\right)\left(v_{max} - v_{ij}\right)$$

Kappa ponderada y kappa

Podemos considerar a kappa como un caso especial de kappa ponderada en donde los $k(k\text{-}1)$ pesos que no pertenecen a celdas de la diagonal principal tienen un valor constante mayor que el valor que tienen las celdas de la diagonal principal (que normalmente será cero para pesos de desacuerdo). Si se da esta situación es fácil ver como la ecuación que define a kappa ponderada equ. 6.16 se simplifica en la ecuación que define a kappa equ. 6.12.

Características muestrales

El error asintótico de kappa ponderada se define como:

$$\sigma_{k_w} = \sqrt{\frac{\sum v_{ij}^2 p_{oij} - \left(\sum v_{ij} p_{oij}\right)^2}{N\left(\sum v_{ij} p_{cij}\right)^2}}$$

equ. 6.21

Como la distribución de kappa ponderada es aproximadamente normal para muestras grandes, se pueden fijar intervalos de confianza utilizando κ_w y $\sigma_{\kappa w}$ de forma similar a como hacíamos en kappa.

También para comprobar la significación de κ_w cuando la κ_w poblacional es cero podemos utilizar la siguiente ecuación:

$$\sigma_{k_w 0} = \sqrt{\frac{\sum v_{ij}^2 p_{cij} - \left(\sum v_{ij} p_{cij}\right)^2}{N\left(\sum v_{ij} p_{cij}\right)^2}}$$

equ. 6.22

Normalizando el valor de κ_w al valor de una normal (0, 1), dividiéndolo por $\sigma_{\kappa w 0}$, nos permite obtener un valor z que podemos analizar en las tablas de la normal.

6.1.3. Medidas de asociación

Las medidas vistas hasta ahora son medidas de acuerdo en el que se trata de cuantificar la fiabilidad o reproducibilidad de las observaciones realizadas. Sin embargo existen otro tipo de medidas, como son las medidas de asociación, en las que lo que se investiga es la relación lineal existente entre las variables.

La asociación entre dos variables (en nuestro caso interpretaciones de expertos) se puede observar fácilmente a partir de diagramas de dispersión (Figura 6.3) que se obtienen representando cada asociación bidimensional (x_i, y_i) como un punto en el plano cartesiano.

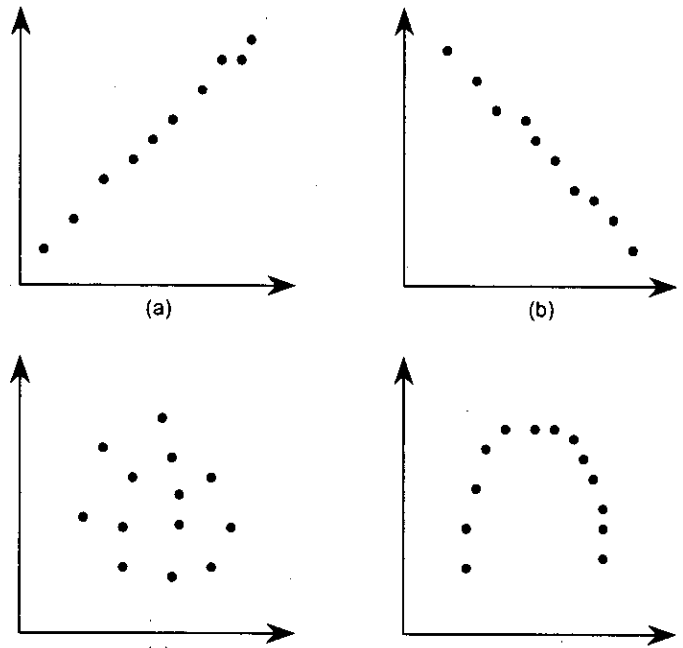


Figura 6.3. Distintos tipos de asociaciones entre variables: (a) asociación lineal positiva, (b) asociación lineal negativa, (c) sin asociación y (d) sin asociación lineal pero con otro tipo de asociación.

Para la medida de la asociación lineal existente entre dos variables se han desarrollado diversas medidas que pasaremos a analizar a continuación.

6.1.3.1. Covarianza entre dos variables

Si x e y son variables aleatorias su covarianza, en cierto sentido, refleja la dirección y la cantidad de asociación o correspondencia entre las variables. La covarianza es grande y positiva si hay una gran probabilidad de que valores grandes (pequeños) de x se asocien con valores grandes (pequeños) de y. Por otro lado, si la correspondencia es inversa de forma que los valores grandes (pequeños) de x ocurren normalmente en conjunción con valores pequeños (grandes) de y, la covarianza tiene un valor grande y negativo.

La covarianza se define según la siguiente ecuación:

$$Cov(x,y)=\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{n}$$

equ. 6.23

en la que n representa en número de muestras que tenemos de cada variable, $\bar{x}$ la media de los valores de x e $\bar{y}$ la media de los valores de y.

El problema que presenta la covarianza como medida de asociación es que su interpretación es complicada ya que su valor depende del orden de magnitud y de las unidades de medida de las variables aleatorias consideradas.

6.1.3.2. El coeficiente de correlación lineal

Este coeficiente nos mide la asociación lineal entre dos variables independientemente de la escala de medida. El coeficiente de correlación se define normalizando la covarianza de ambas variables con las desviaciones típicas de x e y (S_x, S_y) como se muestra en la siguiente ecuación:

$$r = \frac{Cov(x, y)}{S_x S_y}$$

equ. 6.24

Recordemos que la desviación típica de una variable se define como

$$S_x = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n}}$$

equ. 6.25

El coeficiente r (también conocido como el coeficiente de correlación producto-momento de Pearson) se caracteriza porque su valor absoluto no excede de uno, y porque su signo (que viene determinado por el valor de la covarianza) indica si la relación entre las variables es directa o inversa.

Si x e y son variables independientes su correlación es cero, por lo la magnitud de r refleja en cierto sentido el grado de asociación de las variables. Aunque en general no se cumple que una correlación igual a cero implique independencia, la distribución normal bivariable es una excepción. Esta característica puede emplearse para justificar el uso de r como medida de asociación, aunque pierde su importancia cuando tratamos problemas de estadística no paramétrica (en los que no se tiene en cuenta la distribución de la población estudiada). Así cualquier otro tipo de medidas de asociación pueden considerarse perfectamente aceptables pero, debido a que r es perfectamente conocida y usada por todo el mundo, cualquier otra medida de asociación que utilicemos debe emular sus propiedades.

6.1.3.3. Características "adecuadas" de una medida de asociación.

Existen una serie de características que se consideran "adecuadas" para toda medida de asociación. Con estas características los que se pretende es que dichas medidas se asemejen lo más posible al comportamiento del coeficiente de correlación lineal (Gibbons y Chakraborti, 1992).

1. Dados dos pares independientes (x_i, y_i) y (x_j, y_j) de variables aleatorias que siguen una distribución bivariable, la medida debe equivaler a +1 si la relación es directa y perfecta (**acuerdo o concordancia perfecta**) en el sentido de que

   $x_i < x_j$ siempre que $y_i < y_j$, o $x_i > x_j$ siempre que $y_i > y_j$.

2. Dados dos pares independientes (x_i, y_i) y (x_j, y_j) de variables aleatorias que siguen una distribución bivariable, la medida debe equivaler a -1 si la relación es indirecta y perfecta (**desacuerdo o disconcordancia perfecta**) en el sentido de que

$x_i < x_j$ siempre que $y_i > y_j$, o $x_i > x_j$ siempre que $y_i < y_j$.

3. Si el criterio 1 ó el 2 no se cumplen para todos los pares, entonces la medida tomará valores en el intervalo (-1, +1). Es deseable que índices de concordancia elevados estén reflejados por valores positivos elevados, y que índices de discordancia elevados estén reflejados por valores negativos elevados.

4. La medida es cero si x e y son independientes.

5. El valor de la medida para (x, y) debe ser el mismo que para (y, x), $(-x, -y)$ y $(-y, -x)$.

6. El valor de la medida para $(-x, y)$ y para $(x, -y)$ debe ser el negativo de la medida para (x, y).

7. La medida debe ser invariable bajo todas las transformaciones de x e y para las cuales se mantiene el orden de magnitud.

El coeficiente de correlación cumple perfectamente las seis primeras características. Sin embargo, aunque r es invariable bajo transformaciones lineales positivas no es invariable a todas las transformaciones que mantienen el orden, lo que es especialmente deseable cuando trabajamos con datos ordinales ya que tienen un carácter cualitativo y lo que importa no es la distancia entre las categorías sino el orden de las mismas.

Las medidas descritas a continuación pretenden cumplir las siete características anteriores y son medidas no paramétricas (porque no realizan ninguna suposición acerca de la distribución de la población estudiada) que miden asociaciones entre rangos (que es el nombre que se le da a aquellas categorías en las que se conoce su orden pero no su valor). Estas medidas han sido incluidas en la metodología que presentaremos en el captiulo 7 aunque sus diferentes características hacen que algunas sean más adecuadas que otras.

6.1.3.4. Tau de Kendall

A pesar de su nombre, este coeficiente ya había sido propuesto a principios de siglo por Fechner (1897), Lipps (1906) y Deuchler (1914); y fue descrito de forma más teórica en los años 20 por Esscher (1924), y Lindeberg (1925) y (1929). Para consultar detalles sobre su historia se puede consultar Kruskal (1958). Sin embargo Maurice Kendall (1938) no sólo redescubrió este coeficiente sino que investigó sus características no paramétricas por las cuales es utilizado hoy en día, lo que justifica su nombre. Una monografía sobre la tau de Kendall en la que se incluyen teoría, ejemplos y bibliografía puede encontrarse en (Kendall y Gibbons, 1990).

De la misma forma que el coeficiente de correlación de Pearson, los valores de la tau de Kendall varían entre –1 y +1 pero la forma de calcular la asociación es diferente. La tau de Kendall se basa en los conceptos de *concordancia* y *discordancia*. Decimos que un par de observaciones (x_i, y_i) y (x_j, y_j), son concordantes cuando cumplen que $y_i < y_j$ cuando $x_i < x_j$ o $y_i > y_j$ cuando $x_i > x_j$, o lo que es los mismo cuando $(x_i - x_j)(y_i - y_j) > 0$. De la misma forma dos pares de observaciones son discordantes

cuando cumplen que $y_i < y_j$ cuando $x_i > x_j$ o $y_i > y_j$ cuando $x_i < x_j$, o lo que es lo mismo cuando $(x_i - x_j)(y_i - y_j) < 0$.

El número de pares concordantes de un grupo de observaciones se denota con la letra C, mientras que el número de pares discordantes se denota con la letra D. En base a esto podemos definir tau como la proporción de pares concordantes menos la proporción de pares discordantes dentro del conjunto total de pares existentes, que es $n(n-1)/2$ siendo n el número total de casos. Esto expresado matemáticamente sería:

$$\tau = \frac{C}{n(n-1)/2} - \frac{D}{n(n-1)/2} = \frac{C-D}{n(n-1)/2} = \frac{2(C-D)}{n(n-1)}$$

equ. 6.26

A partir de esta ecuación y teniendo en cuenta que $C + D = n(n-1)/2$ es fácil comprobar que:

$$\tau = 1 - \frac{4D}{n(n-1)} = \frac{4C}{n(n-1)} - 1$$

equ. 6.27

Supongamos que a dos catadores de vino les mandamos ordenar, según su preferencia, diez tipos distintos de vinos, y sean sus resultados los mostrados en la Tabla 6.9 en la que cada catador a asignado un número de orden a cada uno de los diez vinos:

	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10
A	7	4	3	10	6	2	9	8	1	5
B	5	7	3	10	1	9	6	2	8	4

Tabla 6.9. Preferencias de vinos de dos catadores.

Ahora queremos calcular si existe algún grado de correlación entre las preferencias de los dos catadores, por lo que analizamos todos los posibles pares de casos, y les asignamos el valor +1 si son concordantes, y el valor –1 si son discordantes.

v1-v2	-1																	
v1-v3	+1	v2-v3	+1															
v1-v4	+1	v2-v4	+1	v3-v4	+1													
v1-v5	+1	v2-v5	-1	v3-v5	-1	v4-v5	+1											
v1-v6	-1	v2-v6	-1	v3-v6	-1	v4-v6	+1	v5-v6	-1									
v1-v7	+1	v2-v7	-1	v3-v7	+1	v4-v7	+1	v5-v7	+1	v6-v7	-1							
v1-v8	-1	v2-v8	-1	v3-v8	-1	v4-v8	+1	v5-v8	+1	v6-v8	-1	v7-v8	+1					
v1-v9	-1	v2-v9	-1	v3-v9	-1	v4-v9	+1	v5-v9	-1	v6-v9	+1	v7-v9	-1	v8-v9	-1			
v1-v10	+1	v2-v10	-1	v3-v10	+1	v4-v10	+1	v5-v10	-1	v6-v10	-1	v7-v10	+1	v8-v10	-1	v9-v10	-1	

Tabla 6.10. Cálculo del número de pares concordantes y discordantes para la Tabla 6.9.

De esta forma $C = 21$ y $D = 24$ por lo que, aplicando la equ. 6.26, obtenemos:

$$\tau = \frac{C-D}{n(n-1)/2} = \frac{21-24}{(10\cdot 9)/2} = \frac{-3}{45} = -0.067$$

lo cual indica que prácticamente no existe asociación entre los dos catadores.

Tau en presencia de ligaduras (Tau b de Kendall)

En muchas ocasiones puede ocurrir un caso no contemplado en el ejemplo anterior y es que pueden existir pares ligados. Decimos que un par de observaciones (x_i, y_i) y (x_j, y_j) están ligados cuando $x_i = x_j$ o $y_i = y_j$.

Si queremos hallar el grado de asociación existente entre los resultados del experto A y del sistema experto podemos extraer los datos de la Tabla 6.2 que reproducimos a continuación

Casos	A	B	C	D	SE
1	ALTO	ALTO	ALTO	ALTO	ALTO
2	ALTO	BAJO	ALTO	ALTO	BAJO
3	BAJO	BAJO	NORMAL	NORMAL	BAJO
4	NORMAL	BAJO	NORMAL	NORMAL	NORMAL
5	MUY ALTO	MUY ALTO	ALTO	ALTO	MUY ALTO
6	BAJO	BAJO	BAJO	BAJO	NORMAL
7	MUY BAJO	MUY BAJO	NORMAL	BAJO	BAJO
8	NORMAL	NORMAL	NORMAL	ALTO	NORMAL
9	NORMAL	NORMAL	BAJO	MUY BAJO	NORMAL
10	BAJO	BAJO	BAJO	ALTO	BAJO

Los datos de A y SE se resumen en la Tabla 6.11 siguiendo la siguiente codificación (↓↓ = Muy Bajo, ↓ = Bajo, = = Normal, ↑ = Alto, y ↑↑ = Muy Alto):

	1	2	3	4	5	6	7	8	9	10
A	↑	↑	↓	=	↑↑	↓	↓↓	=	=	↓
SE	↑	↓	↓	=	↑↑	=	↓	=	=	↓

Tabla 6.11. Datos del experto A y del sistema experto extraídos de la Tabla 6.2.

Para operar mejor con estos datos podemos asignarles rangos numéricos (↓↓ = 1, ↓ = 2, etc.). Es conveniente recordar que el valor asignado no tiene importancia, lo que verdaderamente importa es el orden de las categorías. La nueva tabla quedaría de la siguiente forma

	1	2	3	4	5	6	7	8	9	10
A	4	4	2	3	5	2	1	3	3	2
SE	4	2	2	3	5	3	2	3	3	2

Tabla 6.12. Rangos para los datos de la Tabla 6.11.

A continuación hallamos los pares concordantes y discordantes.

1-2	*0*																
1-3	*+1*	2-3	*0*														
1-4	*+1*	2-4	*-1*	3-4	*+1*												
1-5	*+1*	2-5	*+1*	3-5	*+1*	4-5	*+1*										
1-6	*+1*	2-6	*-1*	3-6	*0*	4-6	*0*	5-6	*+1*								
1-7	*+1*	2-7	*0*	3-7	*0*	4-7	*+1*	5-7	*+1*	6-7	*+1*						
1-8	*+1*	2-8	*-1*	3-8	*+1*	4-8	*0*	5-8	*+1*	6-8	*0*	7-8	*+1*				
1-9	*+1*	2-9	*-1*	3-9	*+1*	4-9	*0*	5-9	*+1*	6-9	*0*	7-9	*+1*	8-9	*0*		
1-10	*+1*	2-10	*0*	3-10	*0*	4-10	*+1*	5-10	*+1*	6-10	*0*	7-10	*0*	8-10	*+1*	9-10	*+1*

Tabla 6.13. Cálculo del número de pares concordantes, discordantes y ligados para la Tabla 6.12.

El número de pares concordantes C es 26, el número de pares discordantes D es 4 y el número de ligaduras T (representadas por un cero en la tabla) es de 15. El valor de τ es:

$$\tau = \frac{C-D}{n(n-1)/2} = \frac{26-4}{(10\cdot 9)/2} = \frac{22}{45} = 0.489$$

En este caso vemos cómo existen 15 pares ligados que no son tenidos en cuenta en el numerador de la ecuación que define a τ, pero que sí son tenidos en el denominador, ya que se consideran todos los pares posibles. Esto implica que, en presencia de ligaduras, τ nunca alcanza el valor unidad, aún cuando exista una concordancia perfecta. Además, la relación $C + D = n(n-1)/2$ deja de ser cierta y pasa a ser $C + D + T = n(n-1)/2$, con lo que la igualdad de la equ. 6.27 deja de cumplirse.

El hecho de eliminar las ligaduras puede tener más sentido en el ejemplo de los vinos, en el que pedíamos un ordenamiento y una ligadura constituye un "fallo" a la hora de establecer dicho orden. El sentido de las ligaduras en entornos de validación se discute en el capítulo 7 al analizar la metodología propuesta de validación.

Para corregir el efecto de las ligaduras se corrigió el índice τ para que también las tuviera en cuenta en el denominador. Esto da lugar a una nueva definición del coeficiente, que se conoce como τ_b, asignando el término τ_a a la equ. 6.26 que era la utilizada hasta ahora. En primer lugar es necesario redefinir el coeficiente τ de la siguiente forma:

$$\tau = \frac{2(C-D)}{\sqrt{\sum_{i=1,j=1}^{n}(a_{ij})^2 \sum_{i=1,j=1}^{n}(b_{ij})^2}}$$

equ. 6.28

en donde n es el número total de casos y a_{ij} es una función definida para los valores de x cuyo valor es +1 si $x_i < x_j$ y -1 si $x_i > x_j$. Análogamente se define b_{ij} para los valores de y.

Si no hay ligaduras se cumple que todos los a_{ij}^2 toman valor 1 (al igual que los b_{ij}^2), por lo que el valor de las sumas será el número de posibles pares ij:

$$\sum_{i=1,j=1}^{n}(a_{ij})^2 = \sum_{i=1,j=1}^{n}(b_{ij})^2 = n(n-1)$$

y la equ. 6.28 quedaría reducida a:

$$\tau = \frac{2(C-D)}{\sqrt{(n(n-1))^2}} = \frac{2(C-D)}{n(n-1)}$$

que es la ecuación original de τ.

Si hay ligaduras es necesario tenerlas en cuenta en el denominador de la siguiente forma: supongamos que para x existe una serie de u miembros consecutivos ligados, esto nos dice que existen $u(u-1)$ pares ligados en x. Si extendemos este valor a las g posibles series de valores ligados que existan en x obtenemos que el número total de pares ligados en x es

$$\sum_{i=1}^{g} u(u-1)$$

por lo que en el denominador de τ habría que sustituir

$$\sum_{i=1,j=1}^{n} (a_{ij})^2 = n(n-1) - \sum_{i=1}^{g} u(u-1)$$

Actuando de forma análoga en y obtendríamos la definición de τ_b como

$$\tau_b = \frac{2(C-D)}{\sqrt{\left[n(n-1) - \sum_{i=1}^{g} u(u-1)\right]\left[n(n-1) - \sum_{i=1}^{h} v(v-1)\right]}}$$

equ. 6.29

en donde h el número de series ligadas en y, y v el número de valores ligados en cada serie.

Esta ecuación se puede simplificar si definimos:

$$U = \frac{\sum_{i=1}^{g} u(u-1)}{2}$$

equ. 6.30

$$V = \frac{\sum_{i=1}^{h} v(v-1)}{2}$$

equ. 6.31

aplicando estas definiciones a la equ. 6.29 obtenemos:

$$\tau_b = \frac{C-D}{\sqrt{\left[\frac{n(n-1)}{2} - U\right]\left[\frac{n(n-1)}{2} - V\right]}}$$

equ. 6.32

Para ver esto con más claridad sigamos con el ejemplo de la Tabla 6.12 que reproducimos a continuación:

	1	2	3	4	5	6	7	8	9	10
A	4	4	2	3	5	2	1	3	3	2
SE	4	2	2	3	5	3	2	3	3	2

Los datos del experto A son (4, 4, 2, 3, 5, 2, 1, 3, 3, 2). Ordenándolos obtenemos (1, 2, 2, 2, 3, 3, 3, 4, 4, 5). De esta forma vemos que tenemos tres series de valores ligados (2, 2, 2), (3, 3, 3) y (4, 4) con 3, 3 y 2 valores ligados respectivamente. De aquí podemos calcular U como

$$U = \frac{\sum_{i=1}^{g} u(u-1)}{2} = \frac{3(3-1)+3(3-1)+2(2-1)}{2} = \frac{6+6+2}{2} = \frac{14}{2} = 7$$

Para el sistema experto, una vez ordenados sus valores tenemos (2, 2, 2, 2, 3, 3, 3, 3, 4, 5). Lo que significa que hay dos series de valores ligados (2, 2, 2, 2) y (3, 3, 3, 3) de 4 valores ligados respectivamente. De aquí podemos calcular V como

$$V = \frac{\sum_{i=1}^{h} v(v-1)}{2} = \frac{4(4-1)+4(4-1)}{2} = \frac{12+12}{2} = \frac{24}{2} = 12$$

De esta forma τ_b sería:

$$\tau_b = \frac{C-D}{\sqrt{\left[\frac{n(n-1)}{2}-U\right]\left[\frac{n(n-1)}{2}-V\right]}} = \frac{26-4}{\sqrt{\left[\frac{10(10-1)}{2}-7\right]\left[\frac{10(10-1)}{2}-12\right]}} =$$

$$= \frac{22}{\sqrt{(45-7)(45-12)}} = \frac{22}{\sqrt{38 \cdot 33}} = \frac{22}{35.412} = 0.621$$

Como vemos el valor de τ_b es mayor que el de τ para los mismos datos (τ = 0.489) cuando tenemos ligaduras porque τ_b se encarga de eliminarlas del denominador. En caso de que no haya ligaduras $\tau = \tau_b$.

Tests de significación

El valor de la tau de Kendall puede usarse como estadístico para probar la hipótesis nula H_0 de que x e y son independientes o no tienen asociación. Existen tres posibles hipótesis alternativas:

A_+: Asociación positiva
A_-: Asociación negativa
A : Existe asociación

La distribución muestral de τ bajo H_0 para $n \leq 30$ es obtenida mediante tablas que pueden consultarse en (Kendall y Gibbons, 1990) y (Gibbons, 1993). Para $n > 30$, tau se aproxima a una normal de media cero y desviación típica igual a:

$$\sigma = \frac{\sqrt{2(2n+5)}}{3\sqrt{n(n-1)}}$$

equ. 6.33

Normalizando el valor de tau a una normal estándar obtenemos:

$$z = \frac{3\tau\sqrt{n(n-1)}}{\sqrt{2(2n+5)}}$$

equ. 6.34

Si en la muestra existen empates deberemos usar τ_b en lugar de τ. La distribución nula de τ_b no puede ser expresada de forma general porque depende de la configuración particular de las ligaduras. Si éstas no son muy abundantes pueden seguir usándose las mismas expresiones que τ para hallar σ y z.

Gamma de Goodman-Kruskal

Como hemos visto hasta ahora, el valor absoluto de la τ de Kendall nunca podrá ser la unidad en presencia de ligaduras porque los pares ligados se eliminan del numerador pero no del denominador. La corrección de las ligaduras realizadas en la τ_b de Kendall permite que el valor obtenido sea mayor que τ pero sigue sin cumplirse que cuando exista un acuerdo o un desacuerdo perfecto el valor absoluto de sea τ_b uno. Este efecto se agrava cuando el número de ligaduras es muy elevado (como suele ocurrir cuando tratamos con tablas de contingencia). Por ello Goodman y Kruskal (1954) propusieron el coeficiente gamma (γ), cuyo valor también oscila entre [-1, 1] y que sí que alcanzaba la unidad en situaciones de acuerdo o desacuerdo perfecto. Gamma se define como:

$$\gamma = \frac{C - D}{C + D}$$

equ. 6.35

en donde C y D son, respectivamente, el número de pares concordantes y el número de pares discordantes.

De esta forma se eliminan por completo, tanto del numerador como del denominador, aquellos pares ligados y el valor absoluto de γ es la unidad en caso de acuerdo o desacuerdo perfecto.

Si no se presentan ligaduras el valor de γ es idéntico al de la τ de Kendall. Para tablas de contingencia de 2x2 el índice γ coincide con el índice Q definido por Yule (1912).

Para los datos de los expertos A y SE el valor de γ es

$$\gamma = \frac{C - D}{C + D} = \frac{26 - 4}{26 + 4} = \frac{22}{30} = 0.733$$

6.1.3.5. Rho de Spearman

Este coeficiente fue introducido por el psicólogo Charles Spearman a principios de siglo (Spearman, 1904) y (Spearman, 1906). Se representa habitualmente como r_s (o también como ρ_s) y básicamente es un coeficiente de correlación basado en rangos y no en valores.

En el apartado 6.1.3.2 describíamos el coeficiente de correlación lineal de Pearson como

$$r = \frac{Cov(x, y)}{S_x S_y}$$

Conociendo la definición de la covarianza (equ. 6.23) y la definición de la desviación típica (equ. 6.25) podemos desarrollar los términos de esta ecuación para obtener otra definición equivalente de r

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

equ. 6.36

Sin embargo, habíamos visto que este coeficiente no era adecuado para medir el grado de asociación de datos ordinales porque no era invariable a todas las transformaciones de x e y para las cuales se mantiene el orden de magnitud.

Supongamos ahora que ordenamos los valores de x de menor a mayor y le asignamos al menor valor el rango 1, al siguiente el rango 2 y así sucesivamente hasta asignarle al mayor valor el rango n. Si actuamos de la misma forma en los valores y, los pares (x_i, y_i) de la muestra inicial se han convertido en pares (R_i, S_i), en donde R_i = rango(x_i) y S_i = rango(y_i). La correlación de estos rangos utilizando la equ. 6.36 que define al coeficiente de correlación lineal es lo que se conoce como el coeficiente de correlación de rangos de Spearman y se representa por

$$r_s = \frac{\sum_{i=1}^{n}(R_i - \bar{R})(S_i - \bar{S})}{\sqrt{\sum_{i=1}^{n}(R_i - \bar{R})^2 \sum_{i=1}^{n}(S_i - \bar{S})^2}}$$

equ. 6.37

Como los rangos son invariables ante transformaciones que mantienen el orden, el coeficiente r_s también será invariable ante dichas transformaciones (además de mantener las características propias de r). Esto es importante porque, en nuestro caso, las interpretaciones de los distintos expertos normalmente seguirán un escala ordinal.

En caso de que no halla empates la equ. 6.37 puede reducirse a una expresión más sencilla

$$r_s = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2 - 1)}$$

equ. 6.38

en donde $d_i = R_i - S_i$. La demostración de esta transformación puede encontrarse en (Gibbons y Chakraborti, 1992).

Por ejemplo, supongamos que, sometemos a ocho alumnos a dos exámenes sobre materias distintas (Tabla 6.14).

Alumno	Examen 1	Examen 2
A	80	44
B	72	72
C	60	69
D	78	70
E	53	93
F	61	82
G	98	67
H	74	80

Tabla 6.14. Calificaciones de ocho alumnos en dos exámenes distintos.

Ahora deseamos saber si existe algún grado de asociación entre el orden de alumnos obtenido mediante el examen 1 y el orden de alumnos obtenido mediante el examen 2. Para ello es necesario convertir los valores numéricos en rangos de la siguiente forma:

53	60	61	72	74	78	80	98
"	"	"	"	"	"	"	"
1	2	3	4	5	6	7	8

44	67	69	70	72	80	82	93
"	"	"	"	"	"	"	"
1	2	3	4	5	6	7	8

con lo que los datos de la Tabla 6.14 quedarían convertidos en los siguientes rangos

Experto	Examen 1	Examen 2
A	7	1
B	4	5
C	2	3
D	6	4
E	1	8
F	3	7
G	8	2
H	5	6

Tabla 6.15. Calificaciones de ocho alumnos en dos exámenes distintos (rangos).

Como no hay empates podemos utilizar la equ. 6.38, por lo que deberemos hallar los valores d_i^2

Alumno	R_i	S_i	d_i	d_i^2
A	7	1	6	36
B	4	5	-1	1
C	2	3	-1	1
D	6	4	2	4
E	1	8	-7	49
F	3	7	-4	16
G	8	2	6	36
H	5	6	-1	1
Total				***144***

Tabla 6.16. Cálculo de d_i^2 para los datos de la Tabla 6.15

Sustituyendo el valor de d_i^2 en la ecuación equ. 6.38 obtenemos:

$$r_s = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)} = 1 - \frac{6(144)}{8(64-1)} = 1 - \frac{864}{504} = 1 - 1.714 = -0.714$$

Lo que significa que la relación existente entre el orden de alumnos de los dos exámenes es inversamente proporcional, es decir, si un alumno esta situado entre los primeros en un examen, estará situado entre los últimos en el otro examen (y viceversa).

Rho en presencia de ligaduras

La presencia de ligaduras en las observaciones afecta a la asignación de rangos previa al cálculo del coeficiente r_s (en el capitulo 7 discutiremos sobre la interpretación de las ligaduras de rho en entornos de validación). Dos formas posibles de tratar estas ligaduras son:

1. *Rangos aleatorios*: se asigna aleatoriamente a los valores ligados los rangos que les pertenecen. Esto permite seguir utilizando las mismas ecuaciones que utilizábamos cuando no había ligaduras, pero añade un elemento aleatorio a la medida que hace que su significado sea poco intuitivo.

2. *Rangos medios*: se asigna a cada miembro del grupo ligado el promedio de los rangos que se le habrían asignado de no haber estado ligados. Esto evita incluir aleatoriedad en la medida, y es una forma muy común de actuar para asignar rangos a ligaduras. El problema es que la reducción de r_s realizada en la equ. 6.38 deja de ser válida.

Si utilizamos la solución de rangos medios en presencia de ligaduras podemos simplificar la equ. 6.37 de la siguiente forma

$$r_s = \frac{n^3 - n - 6\sum_{i=1}^{n} d_i^2 - 6(u'+v')}{\sqrt{(n^3 - n - 12u')(n^3 - n - 12v')}}$$

equ. 6.39

en donde $u' = (\Sigma u^3 - \Sigma u)/12$ siendo u el número de observaciones ligadas en cualquier rango para los valores de R_i. Análogamente definimos $v' = (\Sigma v^3 - \Sigma v)/12$ para las ligaduras en S_i. La demostración de esta transformación puede encontrarse en (Gibbons y Chakraborti, 1992).

La equ. 6.39 es equivalente al coeficiente de correlación de rangos de la equ. 6.37 cuando utilizamos rangos medios para resolver los empates. Si no hay empates entonces $u' = v' = 0$ y la equ. 6.39 se puede reducir a la equ. 6.38. Si el número de empates es muy pequeño en comparación con el número de pares se puede demostrar que el valor de la equ. 6.39 es muy similar al de la equ. 6.38.

Por ejemplo, supongamos los datos de los expertos A y SE ya representados en la Tabla 6.11

	1	2	3	4	5	6	7	8	9	10
A	↑	↑	↓	=	↑↑	↓	↓↓	=	=	↓
SE	↑	↓	↓	=	↑↑	=	↓	=	=	↓

Si ordenamos los resultados de menor a mayor obtenemos los siguientes rangos medios para el experto A

	Resultados ordenados del experto A									
Valores	↓↓	↓	↓	↓	=	=	=	↑	↑	↑↑
Rangos correspondientes	1	2	3	4	5	6	7	8	9	10
Rangos medios resultantes	1		3			6			8.5	10

Tabla 6.17. Asignación de rangos medios al experto A.

y los siguientes rangos medios para el experto SE

	Resultados ordenados del experto SE									
Valores	↓	↓	↓	↓	=	=	=	=	↑	↑↑
Rangos correspondientes	1	2	3	4	5	6	7	8	9	10
Rangos medios resultantes		2.5				6.5			9	10

Tabla 6.18. Asignación de rangos medios al experto SE.

Con lo que la Tabla 6.11 quedaría de la siguiente forma

	1	2	3	4	5	6	7	8	9	10
A	8.5	8.5	3	6	10	3	1	6	6	3
SE	9	2.5	2.5	6.5	10	6.5	2.5	6.5	6.5	2.5

Tabla 6.19. Datos de la Tabla 6.11 representados a partir de rangos medios.

Con estos valores podemos hallar el valor de d_i^2

Caso	A	SE	d_i	d_i^2
1	8.5	9.0	-0.5	0.25
2	8.5	2.5	6.0	36.00
3	3.0	2.5	0.5	0.25
4	6.0	6.5	-0.5	0.25
5	10.0	10.0	0.0	0.00
6	3.0	6.5	-3.5	12.25
7	1.0	2.5	-1.5	2.25
8	6.0	6.5	-0.5	0.25
9	6.0	6.5	-0.5	0.25
10	3.0	2.5	0.5	0.25
Total				***52.00***

Tabla 6.20. Cálculo de d_i^2 para los datos de la Tabla 6.19.

Como existen ligaduras es necesario calcular las correcciones u' y v'. En el experto A existen tres conjuntos de valores ligados y en el experto SE dos.

	u (Experto A)	u^3	v (Experto SE)	v^3
	3	27	4	64
	3	27	4	64
	2	8		
Totales	***8***	***62***	***8***	***128***
		$u' = (62-8) / 12 = 4.5$		**$v' = (128-8) / 12 = 10$**

Tabla 6.21. Cálculo de las ligaduras para los datos de la Tabla 6.11.

Aplicando estos valores a la equ. 6.39 obtenemos el siguiente valor de la r_s de Spearman

$$r_s = \frac{n^3 - n - 6\sum_{i=1}^{n} d_i^2 - 6(u'+v')}{\sqrt{(n^3 - n - 12u')(n^3 - n - 12v')}} = \frac{1000 - 10 - 6(52) - 6(14.5)}{\sqrt{[1000 - 10 - 12(4.5)][1000 - 10 - 12(10)]}} =$$

$$= \frac{591}{\sqrt{(936)(870)}} = \frac{591}{902.397} = .655$$

Lo que significa que existe una relación lineal positiva entre los resultados de los dos expertos, aunque su valor no está muy cercano a la asociación perfecta.

Tests de significación

De la misma forma que tau, el valor de la Rho de Spearman puede usarse como estadístico para probar la hipótesis nula H_0 de que x e y son independientes o no tienen asociación. Las hipótesis alternativas también podrias ser tres:

A_+: Asociación positiva
A_- : Asociación negativa
A : Existe asociación

La distribución muestral de r_s bajo H_0 para $n \leq 30$ es obtenida mediante tablas que pueden consultarse en (Kendall y Gibbons, 1990) y (Gibbons, 1993). Para $n > 30$, rho se aproxima a una normal de media cero y desviación típica igual a:

$$\sigma = \frac{1}{\sqrt{n-1}}$$

equ. 6.40

Normalizando el valor de rho a una normal estándar obtenemos:

$$z = r_s \sqrt{n-1}$$

equ. 6.41

Si en la muestra existen empates deberemos usar la rho sin simplificar que representamos en la equ. 6.39. La distribución nula de esta rho no puede ser expresada de forma general porque depende de la configuración particular de las ligaduras. Si éstas no son muy abundantes pueden seguir usándose las mismas expresiones que para la r_s sin ligaduras.

6.2. Medidas de grupo

Hay tres clases de mentiras: las mentiras, las malditas mentiras y las Estadísticas
Mark Twain (Escritor, periodista y humorista estadounidense. 1.835 – 1.910)

La estadística es una ciencia que demuestra que si mi vecino tiene dos automóviles y yo ninguno, los dos tenemos un automóvil
George Bernard Shaw (Dramaturgo irlandés. 1.856 – 1.950)

Un hombre con un reloj sabe que hora es. Un hombre con dos relojes nunca está seguro.
Ley de Segal

Un comité es un grupo de personas que, individualmente, no pueden hacer nada, pero como grupo pueden decidir que nada puede hacerse
Fred Allen

Hasta ahora hemos visto cómo medir el acuerdo y/o la asociación entre un par de expertos. Estos "expertos" pueden ser expertos humanos, un sistema experto, la opinión consensuada de un grupo de expertos o incluso, la solución real al problema planteado. Sin embargo, como hemos explicado anteriormente, los estándares son muy difíciles de encontrar cuando tratamos de validar un sistema experto. Lo normal será disponer de las opiniones de varios expertos humanos y de los resultados del sistema experto. Con estos datos trataremos de determinar, a través de diversas medidas de grupo, si los resultados del sistema experto son similares o los de los expertos humanos.

El proceso fundamental de aplicación de las medidas de grupo es el que se muestra en la Figura 6.4:

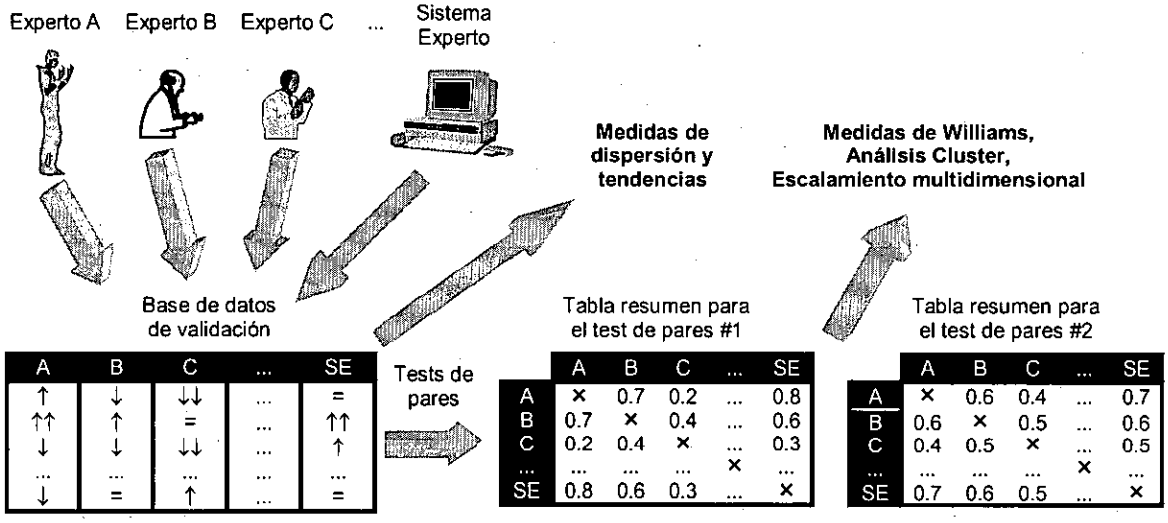


Figura 6.4. Proceso de realización de los tests de grupo.

En esta figura vemos como, en primer lugar se obtiene la base de datos de validación en la que se comparan los resultados de los distintos expertos para una serie de casos seleccionados. Posteriormente se llevan a cabo los tests de pares descritos en el apartado anterior y sus resultados se detallan en una serie de tablas resumen (una por cada test de pares seleccionado), en la que se incluyen los resultados obtenidos por todos los pares posibles de expertos. En base a esta información podemos llevar a cabo distintos tests de grupo como las medidas de Williams, el análisis cluster o el

escalamiento multidimensional. Una excepción a este proceso serían las medidas de dispersión y tendencias que se obtienen directamente de la base de datos de validación.

Veamos ahora con un poco de detalle cada uno de los tests de grupo mencionados.

6.2.1. Medidas de Williams

En la literatura podemos encontrar diversos métodos para medir el acuerdo entre dos o más expertos (o evaluadores) cuyas respuestas son de tipo categórico (ordinales o nominales). Fleiss (1971) generalizó el estadístico kappa propuesto por Cohen (1960) para el caso de una muestra de N casos que es evaluada por n expertos. Light (1971) también consideró el problema de medir el acuerdo entre dos o más expertos y propuso otra medida que también es una extensión de la medida kappa.

Sin embargo, en muchas situaciones podemos no estar interesados en comparar la consistencia interna de dos o más expertos. En vez de ello, podemos aislar un determinado experto y comparar las respuestas del experto aislado con las respuestas del resto del grupo. Light (1971) consideró el problema de comparar el acuerdo de varios expertos con un conjunto "correcto" de respuestas.

Williams (1976) trató un problema similar al de Light pero desde una óptica completamente distinta. Si suponemos que no disponemos de un conjunto "correcto" de respuestas, el objetivo de Williams es decidir si, dado un grupo de expertos y un experto aislado, el acuerdo entre el experto aislado y el grupo era similar al acuerdo existente entre dos miembros cualesquiera del grupo. Para ello define la medida I_n que describimos a continuación.

Sea N el número total de casos a considerar, $n + 1$ el número de expertos y k el número de categorías sobre las cuales se hacen los diagnósticos. Asumimos que cada uno de los $n + 1$ expertos evalúan independientemente los N casos y les asignan una de las k categorías. Definimos $m_{i_0 i_1 \ldots i_n}$ como el número de casos que han sido asignados por el experto 0 a la categoría i_0, por el experto 1 a la categoría i_1, ..., y por el experto n a la categoría i_n. Así, si consideramos cuatro expertos, el valor m_{1111} representa el número de veces que los cuatro expertos han coincidido con el diagnóstico de la categoría 1.

Consideremos el porcentaje de acuerdo observado (ya visto en el apartado 6.1.2.1) y que Williams, redefine siguiendo su notación, como:

$$P_{(a,b)} = \frac{\sum_{i_a=1, i_b=1}^{k} m_{i_a i_b}}{N} \qquad i_a = i_b$$

equ. 6.42

Basándonos en este coeficiente podemos definir el acuerdo existente dentro del grupo de expertos como la media de los acuerdos existentes entre los posibles pares de expertos que pertenecen a dicho grupo (denotado como P_n):

$$P_n = 2\frac{\sum_{a=1}^{n-1}\sum_{b=a+1}^{n} P_{(a,b)}}{n(n-1)}$$

equ. 6.43

El acuerdo total del experto aislado con el grupo de referencia puede ser medido por la media de los distintos $P_{(0,\,a)}$, siendo a = 1, ..., n, de la siguiente forma:

$$P_0 = \frac{\sum_{a=1}^{n} P_{(0,a)}}{n}$$

equ. 6.44

De esta forma podemos definir el índice

$$I_0 = \frac{P_0}{P_n}$$

equ. 6.45

como el índice que nos relaciona la equ. 6.43 y la equ. 6.44. La interpretación de este índice es la siguiente:

- Si I_n es menor que uno indica que el experto aislado está más en desacuerdo con el conjunto de expertos de lo que lo están los miembros del grupo entre sí.
- Si I_n es igual a uno indica que el experto aislado está de acuerdo con el conjunto de expertos tan a menudo como lo están los miembros del grupo entre sí.
- Si I_n es mayor que uno indica que el experto aislado está de acuerdo con el conjunto de expertos en una relación mayor a como lo están los miembros del grupo entre si.

El índice I_n trata de la misma forma a todos los desacuerdos (tal y como pasaba con el índice kappa). En determinadas escalas nominales, y en las escalas ordinales, puede resultar más adecuado establecer una ponderación entre los distintos desacuerdos, tal y como hacía Cohen con el coeficiente kappa ponderada.

Williams propone la utilización de pesos de acuerdo de forma que, se le asigne el máximo peso a las casillas que correspondan a un acuerdo total, mientras que los distintos tipos de desacuerdo se escalan de forma decreciente según va aumentando el grado de desacuerdo. Aunque no es necesario, se recomienda el uso del cero como peso mínimo porque de esta forma denotamos que el acuerdo es nulo en esa categoría.

La proporción de acuerdo ponderado se puede calcular de la siguiente forma:

$$P^{*}_{(a,b)} = \frac{\sum_{i_a=1,\, i_b=1}^{k} w_{i_a i_b} m_{i_a i_b}}{N}$$

equ. 6.46

en donde $w_{i_a i_b}$ representa el peso designado para el caso en el que el experto a selecciona la categoría i_a y el experto b la categoría i_b.

Sustituyendo el porcentaje de acuerdo por el porcentaje de acuerdo ponderado en la equ. 6.43 y la equ. 6.44 obtenemos los valores P_n^* y P_0^* que, sustituidos a su vez en la equ. 6.45, nos permite obtener un nuevo índice I_n^* con la misma interpretación que I_n pero utilizando en porcentaje de acuerdo ponderado.

El índice I_n^* es invariable respecto a cualquier transformación multiplicativa de los índices $w_{i_a i_b}$ (que sea mayor de cero).

El articulo original de Williams se limitaba a utilizar la medida I_n en base al porcentaje de acuerdo y al porcentaje de acuerdo ponderado. Sin embargo, no existe ninguna limitación que impida extender este índice a otras medidas de acuerdo (como los índices kappa o el porcentaje de acuerdo dentro de uno) o asociación (como tau, gamma y rho). De esta forma podríamos comprobar si el acuerdo después de haber corregido la casualidad o el grado de asociación entre el experto aislado y el grupo de expertos, es similar al existente dentro de dicho grupo.

Veamos un ejemplo, recordemos la base de datos de validación incluida cuando describíamos las medidas de pares y que reproducimos a continuación (Tabla 6.2). Esta base de datos contienen las interpretaciones de cuatro expertos humanos y un sistema experto sobre un determinado diagnóstico dividido en cinco etiquetas semánticas.

Casos	A	B	C	D	SE
1	ALTO	ALTO	ALTO	ALTO	ALTO
2	ALTO	BAJO	ALTO	ALTO	BAJO
3	BAJO	BAJO	NORMAL	NORMAL	BAJO
4	NORMAL	BAJO	NORMAL	NORMAL	NORMAL
5	MUY ALTO	MUY ALTO	ALTO	ALTO	MUY ALTO
6	BAJO	BAJO	BAJO	BAJO	NORMAL
7	MUY BAJO	MUY BAJO	NORMAL	BAJO	BAJO
8	NORMAL	NORMAL	NORMAL	ALTO	NORMAL
9	NORMAL	NORMAL	BAJO	MUY BAJO	NORMAL
10	BAJO	BAJO	BAJO	ALTO	BAJO

En base a esta tabla se han calculado los porcentajes de acuerdo y el coeficiente rho de Spearman para todos los posibles pares de expertos. Los resultados se muestran en la Tabla 6.22 y en la Tabla 6.23, respectivamente.

	A	B	C	D	SE
A	—	0.8	0.6	0.4	0.7
B	0.8	—	0.4	0.2	0.7
C	0.6	0.4	—	0.6	0.4
D	0.4	0.2	0.6	—	0.3
SE	0.7	0.7	0.4	0.3	—

Tabla 6.22. Tabla resumen para el porcentaje de acuerdo.

	A	B	C	D	SE
A	—	0.787	0.693	0.562	0.655
B	0.787	—	0.361	0.411	0.837
C	0.693	0.361	—	0.604	0.288
D	0.562	0.411	0.604	—	0.153
SE	0.655	0.837	0.288	0.153	—

Tabla 6.23. Tabla resumen para la rho de Spearman.

Aplicando las medidas de Williams a los datos de la Tabla 6.22 (tomando como experto aislado al sistema experto – SE –) obtenemos el siguiente I_n para el porcentaje de acuerdo:

$$P_n = 2\frac{\sum_{a=1}^{n-1}\sum_{b=a+1}^{n} P_{(a,b)}}{n(n-1)} = \frac{2(0.8+0.6+0.4+0.4+0.2+0.6)}{4\cdot 3} = \frac{6}{12} = 0.5$$

$$P_0 = \frac{\sum_{a=1}^{n} P_{(0,a)}}{n} = \frac{(0.7+0.7+0.4+0.3)}{4} = \frac{2.1}{4} = 0.525$$

$$I_0 = \frac{P_0}{P_n} = \frac{0.525}{0.5} = 1.05$$

y para la r_s de Spearman obtenemos:

$$P_n = \frac{2(0.787+0.693+0.562+0.361+0.411+0.604)}{4\cdot 3} = \frac{6.836}{12} = 0.5697$$

$$P_0 = \frac{(0.655+0.837+0.288+0.153)}{4} = \frac{1.933}{4} = 0.483$$

$$I_0 = \frac{P_0}{P_n} = \frac{0.483}{0.5697} = 0.848$$

lo cual se puede interpretar como que, para el porcentaje de acuerdo, el acuerdo existente entre el sistema experto y los expertos es similar al existente entre los propios expertos. Sin embargo, para la r_s de Spearman, obtenemos que el grado de asociación entre el sistema experto y los expertos es ligeramente inferior al grado de asociación que existe entre los propios expertos.

6.2.2. Análisis cluster

> *Una antigua clasificación china de los animales*
> Los animales se dividen en (a) aquellos que pertenecen al emperador, (b) los embalsamados, (c) aquellos que son entrenados, (d) lechones, (e) sirenas, (f) animales fabulosos, (g) perros callejeros, (h) aquellos que están incluidos en esta clasificación, (i) aquellos que tiemblan como si estuvieran locos, (j) animales innumerables, (k) aquellos peinados con un peine de pelo fino de camello, (l) otros, (m) aquellos que acaban de romper un jarrón de flores y (n) aquellos que parecen moscas desde lejos.
> *"Otras Inquisiciones"*
> *Jorge Luis Borges (Escritor argentino, 1899 – 1986)*

En las medidas de Williams analizadas en el apartado anterior veíamos como se intentaba determinar si los resultados de un experto aislado eran similares a los resultados de un grupo de expertos tomados como referencia. Un tratamiento parecido sería clasificar a los expertos en grupos según la similitud de sus diagnósticos y comprobar si nuestro experto aislado pertenece (o está cerca) de aquel grupo formado por los expertos de mayor experiencia.

El término "análisis cluster" es el nombre genérico asignado a un conjunto de procedimientos que pueden utilizarse para crear una clasificación. De forma más específica, un método de clustering es un procedimiento estadístico multivariable que comienza con un conjunto de datos que contienen información sobre una serie de sujetos y que intenta reorganizar dichos sujetos en grupos o clusters relativamente homogéneos (Aldenderfer y Blashfield, 1984).

La idea de agrupar en clases objetos que comparten características no es nueva, y ha significado un papel importante en muchas ciencias. Así, la clasificación de animales y plantas sentó las bases de la posterior teoría de la evolución de *Darwin*; la clasificación de los elementos químicos en la tabla periódica realizada por Mendeleyev en 1860 tuvo un profundo impacto en la comprensión de la estructura del átomo; y la clasificación de las estrellas usando el diagrama de Hertsprung-Rusell (que relaciona temperatura y luminosidad) ha influido de forma considerable en las teorías sobre la evolución de las mismas.

El comienzo de los procedimientos formales de clasificación puede encontrarse en los trabajos de Kulczynski (1928), Zubin (1938) y Tryon (1939). Pero el mayor estímulo para el desarrollo de métodos de clustering fue el libro titulado *Principles of Numerical Taxonomy*, publicado en 1963 por los biólogos Robert Sokal y Peter Sneath (Sokal y Sneath, 1963). En este libro se propone un nuevo sistema de clasificación biológica que abarca la colección de los datos, la selección y el codificado de las características, el cálculo de la similaridad, la construcción de clusters jerárquicos y su evaluación a partir de técnicas estadísticas. En palabras de Sokal y Sneath "la clasificación es uno de los procesos fundamentales de la ciencia, ya que los fenómenos deben ser ordenados para que podamos entenderlos". Otro factor que ha facilitado la popularización de los métodos de cluster es el empleo de potentes computadores que permiten el desarrollo de complejos algoritmos que serían imposibles de realizar hace tan sólo unos años.

6.2.2.1. Definición de cluster.

Hasta el momento las palabras cluster, grupo y clase han sido utilizadas de manera intuitiva, sin intentar dar una definición formal. Esto se ha hecho así porque no existe una definición de cluster que sea aceptada universalmente (Dubes, 1993). Bonner (1964) sugiere que el criterio final del término debe quedar en manos de cada usuario particular, sin embargo, Cormack (1971) y Gordon (1980) intentan definir el cluster usando características como la *cohesión interna* y el *aislamiento externo*. Así en la Figura 6.5 los observadores pueden distinguir los distintos clusters sin necesitar una definición formal de los mismos, también se puede ver que una simple definición no es suficiente para abarcar todos los casos.

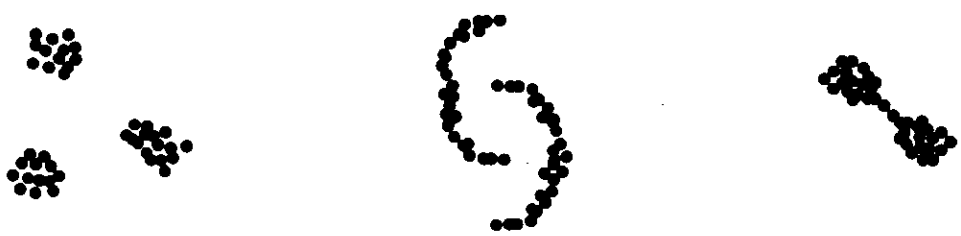


Figura 6.5. Clusters con cohesión interna y aislamiento externo.

En estos ejemplos se observa que una característica fundamental para la determinación de un cluster es la distancia relativa entre los elementos que lo componen. Las distancias son un aspecto muy importante a tener en cuenta cuando desarrollamos un análisis cluster (Everitt, 1993).

Los ejemplos vistos son muy sencillos, sin embargo en el mundo real el investigador no conoce *a priori* la estructura de los datos, entonces se corre el peligro de llegar a interpretaciones suponiendo la existencia de distintos clusters cuando en realidad no es así.

6.2.2.2. Definición de análisis cluster.

El análisis cluster se puede definir como el estudio formal de métodos y algoritmos para organizar objetivamente datos numéricos (Dubes, 1993). Como ya habíamos dicho, en el análisis cluster el investigador no tiene conocimiento *a priori* sobre la estructura subyacente de los datos (o por lo menos es un conocimiento limitado). Esto permite definir claramente el análisis cluster del *análisis discriminante*. En este último el investigador sí conoce *a priori* el esquema de clasificación, y lo que intenta es dilucidar las reglas que determinan la pertenencia de un elemento a un grupo basándose en una serie de variables. Para ello suele disponer de datos de entrenamiento para cada uno de los grupos especificados (McLachlan, 1992).

El análisis cluster también puede verse como una técnica de reducción de datos. Los elementos disponibles se agrupan en clusters de tal forma que los perfiles de los individuos que están en un mismo cluster son similares, mientras que los perfiles de los individuos que están en clusters diferentes presentan bastantes diferencias (Jobson, 1992).

Al realizar un análisis cluster es importante seguir los siguientes pasos (Bisquerra, 1989) y (Dubes, 1993):

1) Selección de las variables relevantes para la identificación de los distintos grupos.
2) Selección de la medida de similitud entre los distintos elementos.
3) Selección del criterio para agrupar individuos en clusters.
4) Validación e interpretación de la estructura obtenida.

6.2.2.3. Variables relevantes.

Es evidente que la elección de las variables que describen los elementos que se pretenden agrupar es fundamental para la realización de un análisis cluster correcto. El primer aspecto que hay que tener en cuenta a la hora de elegir las variables es si éstas son relevantes, o no, para la clasificación que se está llevando a cabo. Por ejemplo, en un análisis que pretende agrupar enfermedades mentales puede no ser adecuado la inclusión de las características físicas de las personas (altura, peso, etc.).

El siguiente aspecto a tener en cuenta es el número de variables que deben ser medidas en cada individuo. En la mayoría de las aplicaciones existen incontables variables que pueden ser utilizadas, sin embargo, consideraciones de tipo temporal o económico suelen restringir su número. Un error que se comete normalmente es el de

considerar más variables de las estrictamente necesarias. Esto puede dar lugar a problemas computacionales en las técnicas de clustering o, más importante, pueden obscurecer la estructura de clusters (Everitt, 1993).

Una vez se han elegido las variables se construye lo que se denomina una *matriz de elementos* o de patrones. Esta matriz tiene dimensiones $n \times d$, en donde n es el número de individuos y d es el número de variables consideradas.

6.2.2.4. Medidas de similitud.

Después de haber determinado las características a tener en cuenta en los individuos que queremos agrupar, es necesario definir un medida de similitud que nos permita medir la proximidad entre dos individuos cualesquiera. Aunque pueda parecer algo simple, el concepto de similitud, y especialmente, los procedimientos establecidos para medir dicha similitud, no son triviales.

La similitud entre dos individuos i y j se define como una función $s_{ij} = f(\mathbf{x}_i, \mathbf{x}_j)$ en donde $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{id}]$ es el vector que representa los d valores asociados al individuo i. La similitud suele ser una relación simétrica ($s_{ij} = s_{ji}$), representada generalmente por coeficientes no negativos y limitados por la unidad ($0 \leq s_{ij} \leq 1$). Sin embargo, algunos presentan características similares a las del coeficiente de correlación ($-1 \leq s_{ij} \leq 1$) y también existen coeficientes no limitados por ningún valor.

Asociados a los coeficientes limitados en el intervalo [0, 1] podemos definir una medida de disimilitud como $d_{ij} = 1 - s_{ij}$. Las principales características que suelen cumplir las disimilitudes son:

1) $0 \leq d_{ij}$ (generalmente con un límite superior igual a la unidad)

2) $d_{ij} = d_{ji}$

3) $d_{ii} = 0$ (o de forma más general $d_{ij} = 0$ sii $\mathbf{x}_i = \mathbf{x}_j$)

Existe una condición más restrictiva, denominada "desigualdad triangular" que establece que $d_{ij} \leq d_{ik} + d_{kj}$. Obviamente se llama desigualdad triangular porque establece que un lado del triángulo que une a los tres elementos es menor o igual que la suma de los otros dos lados. Aquellas disimilitudes que cumplen también la desigualdad triangular se denominan "distancias métricas" o simplemente "métricas".

La desigualdad triangular es una propiedad necesaria, aunque no suficiente, para que los n individuos puedan ser representados como n puntos en un espacio métrico, normalmente euclídeo (esto se verá con más detalle al describir el escalamiento multidimensional). En el apartado 7.1.2 ya hemos descrito algunos tipos de distancias, de los cuales el más popular es la distancia euclídea.

Las similitudes o disimilitudes se suelen ordenar en una matriz $n \times n$ en donde cada par de valores (i, j) contendrá la similitud o la disimilitud entre los elementos i y j. Esta matriz es posteriormente utilizada como entrada de muchos procedimientos de análisis cluster como veremos en apartados posteriores.

Un problema importante a la hora de tratar las similitudes o las distancias surge cuando las distintas variables que describen a un individuo no utilizan las mismas unidades de medida o no son del mismo tipo. En el primer caso la solución que se buscó fue la estandarización de los valores (como la normalización *z* que consiste en restar a los valores la media y dividir el resultado por la desviación típica). Sin embargo Fleiss y Zubin (1969) mostraron que la normalización puede provocar que se diluyan las diferencias entre grupos para variables que mostraban una buena capacidad discriminatoria. Una discusión sobre los efectos de la normalización en el análisis cluster puede encontrarse en (Milligan y Cooper, 1988).

Cuando las variables son de distinto tipo se han sugerido varias soluciones. La más simple es convertir todas las variables a formato binario antes de calcular las similitudes. Por ejemplo la edad puede convertirse en "menor de 40" y "40 o más". Este procedimiento es sencillo pero tiene el inconveniente de que puede sacrificar información útil. Otra posible solución es utilizar un coeficiente que incorpore la información de los distintos tipos de variables de forma adecuada. Un ejemplo de este tipo de coeficiente fue propuesto por Gower (1971).

6.2.2.5. Tipos de análisis cluster

Una vez que hemos decidido cuales son las variables a utilizar en nuestro estudio y hemos elegido una medida de proximidad entre los distintos elementos, el siguiente paso a realizar es elegir el criterio a seguir para agrupar a los individuos en clusters.

El análisis cluster puede clasificarse de muchas maneras atendiendo a distintos criterios, de todas formas existen dos grandes categorías como son los métodos jerárquicos y los métodos no jerárquicos.

Los **métodos jerárquicos** consisten en ir formando grupos en pasos sucesivos, construyendo lo que se conoce como árbol de partición o *dendrograma*, que permite visualizar los resultados. Cada nivel de agregación del dendrograma representa un partición del conjunto dado de individuos. Así en la Figura 6.6 podemos ver un dendrograma de clustering de 6 individuos. Si particionamos el árbol en el nivel X obtenemos los clusters cuya distancia entre cualquier par de expertos dentro de un mismo cluster son menores que X (en este caso obtenemos A-B, C y D-E-F). Variando X obtenemos distintas particiones del conjunto dado.

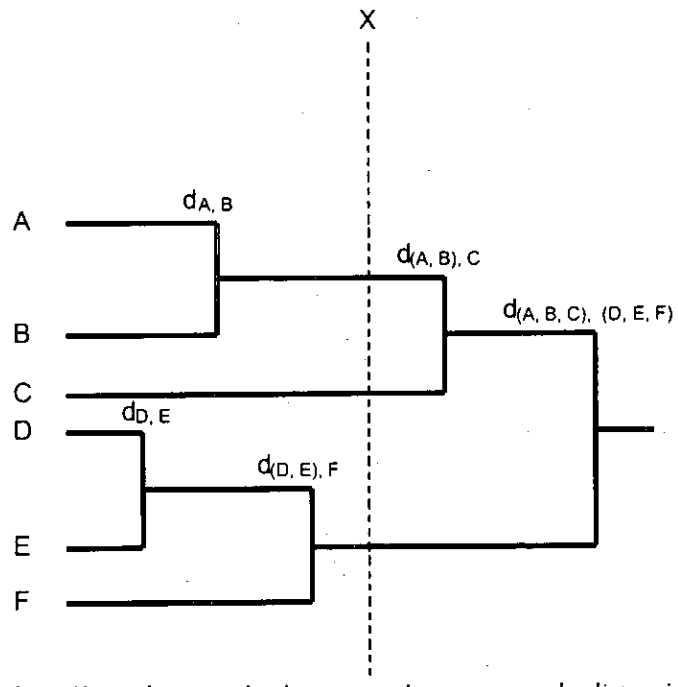


Figura 6.6. Árbol de análisis cluster o dendrograma. d_{ij} representa la distancia entre el punto i y el punto j.

De esta forma hemos obtenido una secuencia completa de soluciones del análisis cluster, empezando con *n* clusters (uno por cada individuo) y terminando con un solo cluster que contiene a los *n* individuos. En algunas aplicaciones se requiere la secuencia entera de soluciones mientras que en otras sólo se requiere una de las posibles soluciones.

Los métodos **no jerárquicos**, también conocidos como métodos iterativos, tienen por objetivo realizar una sola partición de los individuos en *k* clusters. Generalmente buscan aquella partición que optimice un cierto criterio reseñado, moviendo los elementos de un cluster a otro hasta que se encuentra la configuración óptima. Normalmente, como el número de iteraciones necesarias para alcanzar la condición suele ser bastante alto, en la práctica uno mismo se limita para aplicar criterios de optimización locales.

Las diferencias fundamentales entre los métodos jerárquicos y no jerárquicos son las siguientes:

1) En los no jerárquicos el numero de clusters debe especificarse a priori, o calcularse mediante el mismo algoritmo.

2) En los métodos jerárquicos cuando un elemento se asigna a un cluster no puede volver a ser reasignado. Según Kaufman y Rousseeuw (1990) esto puede ser un error ya que "el método jerárquico nunca puede reparar lo que ha hecho en un paso anterior". En los métodos no jerárquicos los elementos puede cambiar de cluster en las distintas iteraciones. Para intentar mejorar los métodos jerárquicos últimamente se está intentando utilizar en su construcción métodos que atiendan a criterios globales y no sólo a criterios locales. Por ejemplo, Lozano y Larrañaga (1998) utilizan métodos globales basados en algoritmos genéticos.

3) Los métodos jerárquicos trabajan con la matriz de similitudes mientras que los no jerárquicos trabajan con la matriz de elementos.

Los métodos jerárquicos pueden a su vez subdividirse en métodos aglomerativos y métodos disociativos. Los métodos **aglomerativos**, también conocidos como ascendentes, empiezan el análisis con tantos grupos como individuos haya. A partir de estas unidades iniciales se van formando grupos de forma ascendente, agrupando cada vez más individuos en los sucesivos grupos que se van formando. Al final del proceso todos los casos están englobados en un mismo cluster. Los métodos **disociativos**, también denominados descendentes o divisivos, constituyen el proceso inverso al anterior. Empiezan con un cluster que engloba a todos los individuos. A partir de este gran grupo inicial, de forma descendente y a través de sucesivas divisiones, se van formando grupos cada vez más pequeños. Al final del proceso se tienen tantos grupos como individuos.

Otro tipo de clasificación es la que divide al análisis cluster en monotética o politética (Cuadras, 1991). Una clasificación **monotética** está basada en una característica única (o en unas pocas) que sea muy relevante. Suele ser divisiva, pues los objetos se clasifican en los que tienen la característica y los que no la tienen. Puede dar lugar a clasificaciones poco adecuadas, dada la dificultad de obtener grupos lo bastante homogéneos y naturales (hay pájaros que no vuelan, mamíferos que viven en el agua, etc.). Una clasificación **politética** está basada en un gran número de características, y no exige que todos los elementos de una clase posean todas las características, sino el número suficiente para poder justificar analogías entre miembros de una misma clase. Este tipo de clasificación suele ser aglomerativo.

En las técnicas vistas hasta ahora un elemento sólo podía pertenecer a un único cluster. Últimamente están apareciendo un nuevo tipo de técnicas, denominadas **técnicas difusas**, en las que se permite que un elemento pertenezca a varios clusters con distinto grado de participación.

Por último, resaltar los **métodos graficos** de clustering. Estos métodos no siguen ningún algoritmo para la realización del análisis cluster sino que se basan en la representación de los datos multivariables en un formato gráfico que permita agruparlos fácilmente por su similitud.

Entre los métodos gráficos cabe destacar el método de los *glifos* y los *petroglifos* sugerido por Anderson (1960), el método de las series de Fourier sugerido por Andrews (1972) y el novedoso método de las caras sugerido por Chernoff (1973). Este método consiste en representar cada elemento del estudio como una cara cuyas características (ojos, nariz, boca, etc.) representan a las variables y su tamaño es proporcional al valor de dichas variables. Por ejemplo, si el valor de una variable es alto los ojos de la cara serán grandes y si el valor es bajo los ojos serán pequeños (Figura 6.7). Un resumen de estos métodos gráficos puede encontrarse en (Dillon y Goldstein, 1984).

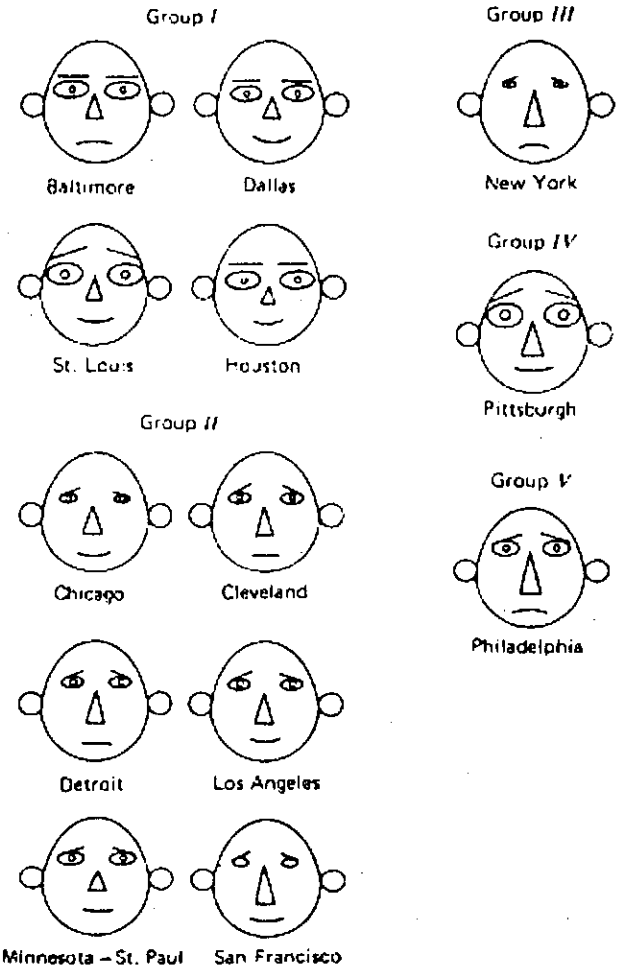


Figura 6.7. Caras de Chernoff para la agrupación de 13 ciudades según sus características economicas. Los grupos se ha obtenido de un análisis cluster jerárquico independiente. Como podemos observar la similitud entre las caras que pertenecen a un mismo grupo es mayor que la similitud entre caras de distintos grupos (obtenido de Huff y Black (1978)).

6.2.2.6. Análisis cluster jerárquico

Los métodos de clustering jerárquico son los más populares y los más utilizados en la bibliografía, además, resultan los más adecuados para utilizar en la validación de sistemas expertos porque permiten utilizar la información que habíamos obtenido en los tests de pares. Por ello, en este trabajo nos centraremos básicamente en las técnicas jerárquicas de clustering.

Estas técnicas parten de la matriz de distancias o similitudes entre los individuos y mediante un algoritmo de clasificación generan una jerarquía indexada que generalmente se representa por un dendrograma. Este proceso puede verse en la Figura 6.8.

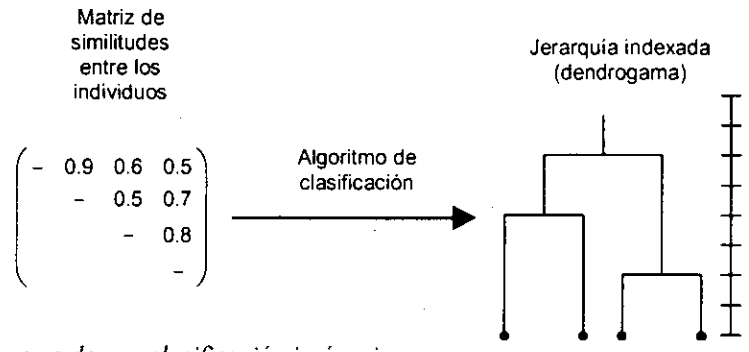


Figura 6.8. Esquema de una clasificación jerárquica.

Hay muchos algoritmos de clasificación jerárquica, y dentro de ellos podemos distinguir los que siguen una tendencia aglomerativa o los que siguen una tendencia disociativa (aunque una vez construido el dendrograma nos es indiferente el método seguido). Los métodos más populares y los más corrientemente utilizados son los aglomerativos, por lo que centraremos nuestro estudio en los mismos. De todas formas los algoritmos aglomerativos descritos también pueden aplicarse a los métodos disociativos, simplemente hay que tener en cuenta que en cada paso se genera un grupo nuevo formado con los casos en los que las distancias son mayores (en vez de ir agrupando paso a paso los elementos más cercanos) .

Análisis cluster jerárquico aglomerativo

En esta sección se estudiarán los algoritmos de clustering que se conocen popularmente por las siglas SAHN (Sequential, Agglomerative, Hierarchical, Nonoverlapping), es decir, algoritmos de clustering secuenciales, aglomerativos, jerárquicos y sin superposición (Dubes, 1993).

Un algoritmo SAHN comienza con una matriz $n \times n$ de distancias entre elementos (siendo n el número de elementos) y con tantos clusters como elementos tengamos. Al final obtendremos una única partición que contendrá a todos los individuos. Antes de describir el proceso conviene aclarar matemáticamente algunos conceptos:

Sea χ el conjunto de n individuos o elementos que disponemos para la realización del análisis cluster. Una *partición* (clustering) $C = \{C_1, C_2, \ldots, C_m\}$ de χ es un conjunto disjunto de subconjuntos no vacíos de χ, que considerados globalmente constituyen χ. Es decir, si $i \neq j$ entonces

$$C_i \cap C_j = \varnothing;$$

$$C_1 \cup C_2 \cup \ldots \cup C_m = \chi$$

Los componentes de una partición se denominan *clusters*. Un *análisis cluster jerárquico* es una secuencia de particiones anidadas que empiezan con la partición trivial, en la cual cada elemento forma un único cluster, y termina en la partición trivial en la cual todos los elementos pertenecen al mismo cluster. La partición $\boldsymbol{B}$ esta anidada dentro de la partición $\boldsymbol{C}$ si cada componente de $\boldsymbol{B}$ es un subconjunto de un componente de $\boldsymbol{C}$, así pues $\boldsymbol{C}$ se forma mezclando componentes de $\boldsymbol{B}$.

Un *dendrograma* es un árbol binario que representa un análisis cluster jerárquico. Cada nodo del árbol representa un cluster y cortando transversalmente el dendrograma obtenemos una partición dada. Un dendrograma representa n particiones de n elementos.

Sea $\{C_0, C_1, \ldots, C_{n-1}\}$ la secuencia de particiones de un dendrograma en el que C_0 es la partición trivial que pone a cada elemento en su propio clustering y C_{n-1} la partición trivial que agrupa a todos los elementos dentro de un mismo cluster. Los clusters de la m-ésima partición se denotan como $\{C_{m1}, C_{m2}, \ldots, C_{m(n-m)}\}$. Se define también una función $L(m)$ en cada partición que representa el *nivel de desigualdad* en el que el clustering m se ha formado.

Como nueva medida de distancia entre los elementos se define la *distancia cofenética* que indica cuál es el primer nivel del dendrograma en el cual dos elementos dados aparecen por primera vez en el mismo cluster. Formalmente la distancia cofenética entre los elementos $\mathbf{x}_i$ y $\mathbf{x}_j$ se define como

$$d_c(i,j) = L(k_{i,j})$$

en donde

$$k_{i,j} = \min [m: (\mathbf{x}_i, \mathbf{x}_j) \in C_{mt}, \text{ para algún } t]$$

La distancia cofenética además de cumplir las características de una distancia cumple la desigualdad ultramétrica de tal forma que

$$d_c(i,j) \leq \max \{d_c(i,k), d_c(k,j)\} \ \forall\ i,j,k$$

Una vez aclarados estos conceptos el algoritmo de clustering SAHN se puede describir de la siguiente forma:

Algoritmo SAHN de clustering jerárquico

1. Fijamos el número de clustering a cero: $m = 0$.

   Repetimos los siguientes pasos hasta que $m = n$.

2. Encontramos el par de clusters i y j cuya distancia es mínima:

   $$d_{ij} = \min \{d_{qr}\} \quad \forall q, r \ / \ 1 \leq q, r \leq n$$

3. Incrementamos m en uno. Mezclamos los clusters i y j en un solo cluster para definir el clustering m. Definimos el nivel de este clustering como:

   $$L(m) = d_{ij}$$

4. Actualizamos la matriz de distancias borrando las filas y columnas que pertenecen a los clusters i y j, y añadiendo una nueva fila y una nueva columna para el cluster recién formando ij. La distancia entre este nuevo cluster y los ya existentes depende del método de clustering que estemos empleando y que veremos a continuación.

En este algoritmo hemos utilizado una matriz de distancias. Si quisiéramos utilizar una matriz de similitudes simplemente habría que cambiar el mínimo del paso 2 por un máximo.

Para ilustrar la aplicación de los distintos métodos de clustering emplearemos los datos provenientes de las coordenadas de distintas ciudades como se muestra en la Tabla 6.24 y en la Tabla 6.25

Ciudad	Latitud	Longitud
A Coruña	43.39	8.38
Zaragoza	41.65	0.90
Madrid	40.40	3.68
Barcelona	41.38	-2.18
Sevilla	37.38	5.98
Valladolid	41.65	4.75

Tabla 6.24. Matriz de elementos para diversas ciudades y sus coordenadas

	A Coruña	Zaragoza	Madrid	Barcelona	Sevilla	Valladolid
A Coruña	—	7.68	5.57	10.75	6.47	4.03
Zaragoza	7.68	—	3.05	3.09	6.64	3.85
Madrid	5.57	3.05	—	5.94	3.80	1.65
Barcelona	10.75	3.09	5.94	—	9.09	6.94
Sevilla	6.47	6.64	3.80	9.09	—	4.44
Valladolid	4.03	3.85	1.65	6.94	4.44	—

Tabla 6.25. Matriz de distancias para los datos de la Tabla 6.24.

Método de las distancias mínimas (single linkage)

Es uno de los métodos más sencillos de clustering jerárquico y también uno de los más utilizados. Se conoce como la técnica del *vecino más cercano* (nearest neighbour) y consiste en definir la distancia entre los nuevos clusters que se van formando como la distancia existente entre sus elementos más próximos. Esta situación puede visualizarse en la Figura 6.9:

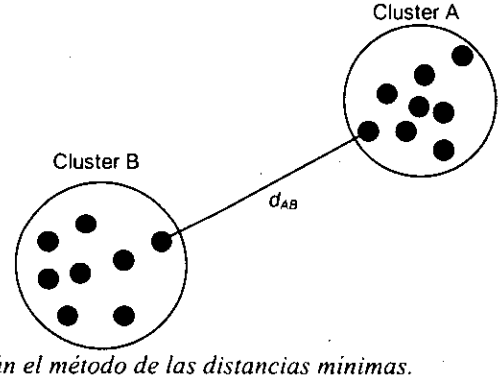


Figura 6.9. Distancia según el método de las distancias mínimas.

Aplicando este algoritmo a los datos de las ciudades obtenemos el resultado de la Figura 6.10.

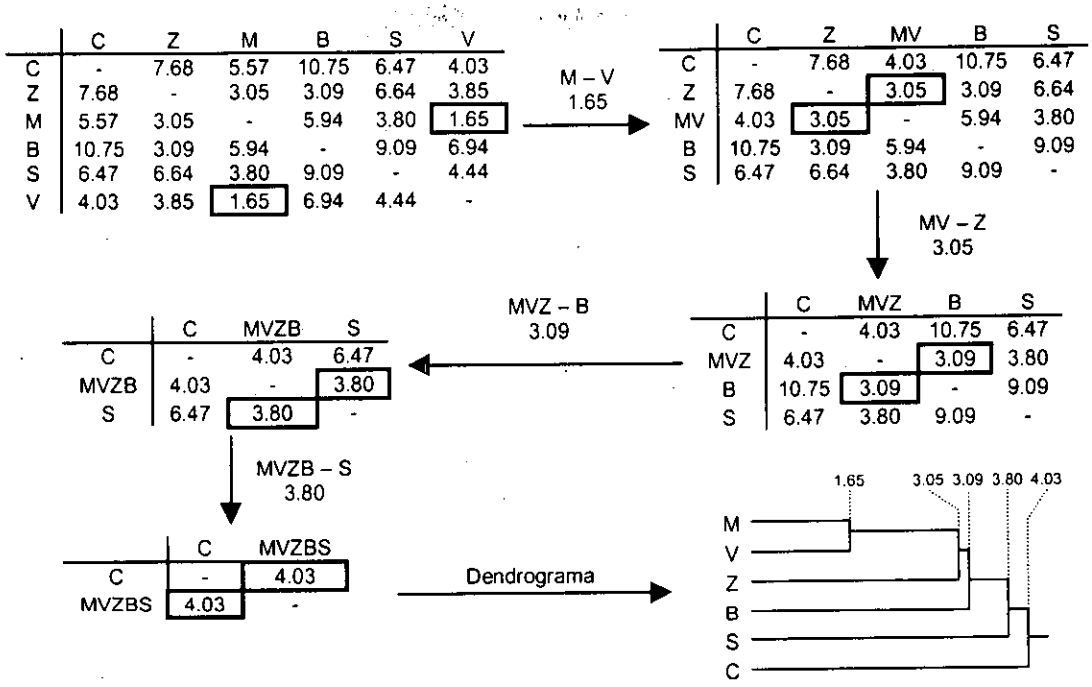


Figura 6.10. Ejemplo de utilización del método de las distancias mínimas utilizando los datos de las ciudades.

Método de las distancias máximas (complete linkage).

Se conoce también como la técnica del vecino mas lejano (furthest neighbour) y representa la técnica contraria al método de las distancias mínimas en el sentido en que la distancia entre dos clusters se define como la distancia existente entre sus elementos más alejados. Este método se ilustra en la Figura 6.11.

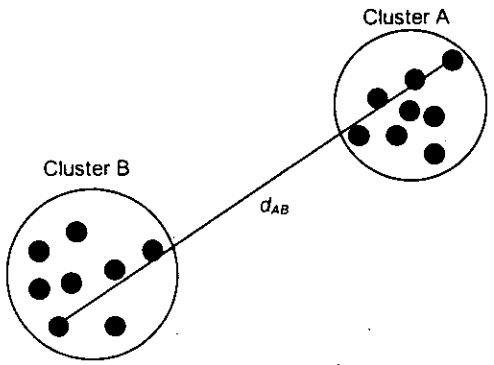


Figura 6.11. Distancia según el método de las distancias máximas.

Aplicando este algoritmo a la matriz de similitudes para el ejemplo de las ciudades obtenemos el resultado de la Figura 6.12.

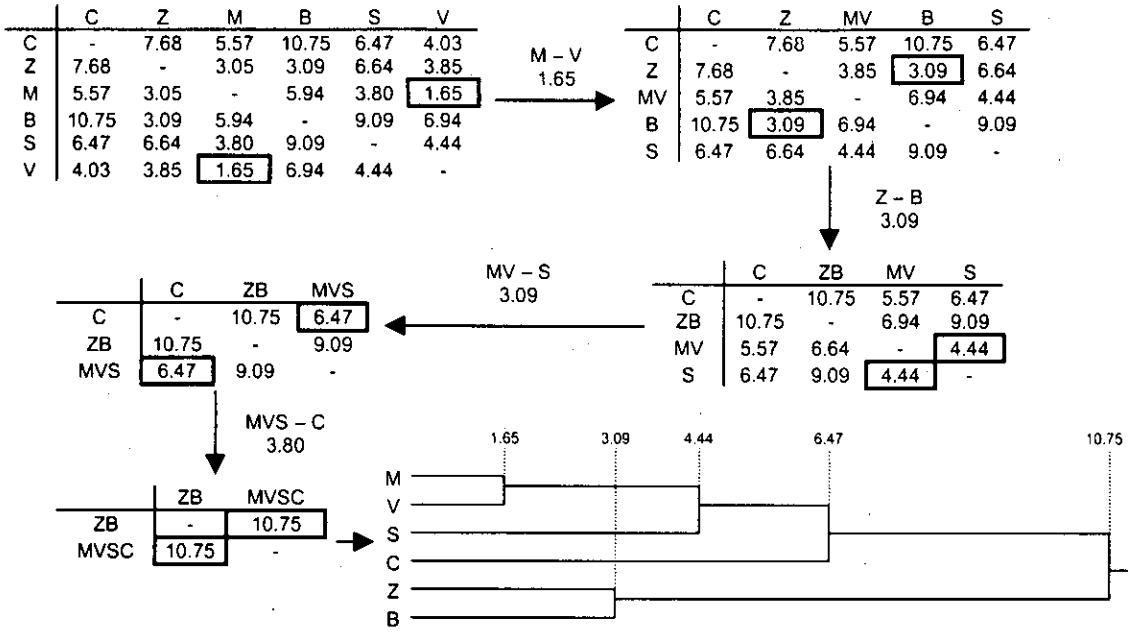


Figura 6.12. Ejemplo de utilización del método de las distancias máximas utilizando los datos de las ciudades.

Método del promedio entre grupos (group average - UPGMA -).

En este método la distancia entre dos clusters se define como la media de las distancias entre todos los pares de individuos en los cuales un miembro del par pertenece a cada uno de los clusters formados anteriormente (Figura 6.13).

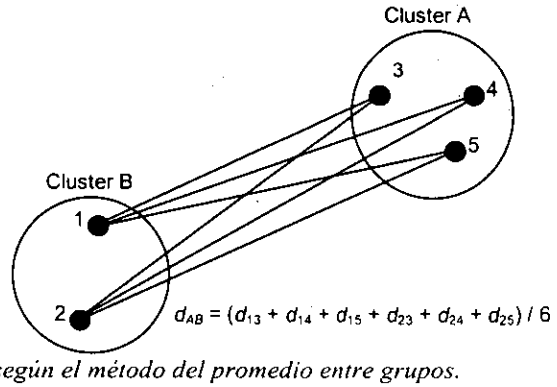


Figura 6.13. Distancia según el método del promedio entre grupos.

Este método utiliza información de todas las distancias entre pares de individuos, y no solamente de los más alejados o de los más próximos, como es el caso de los dos métodos anteriores.

Aplicando este algoritmo a la matriz de similitudes del ejemplo de las ciudades obtenemos el resultado de la Figura 6.14.

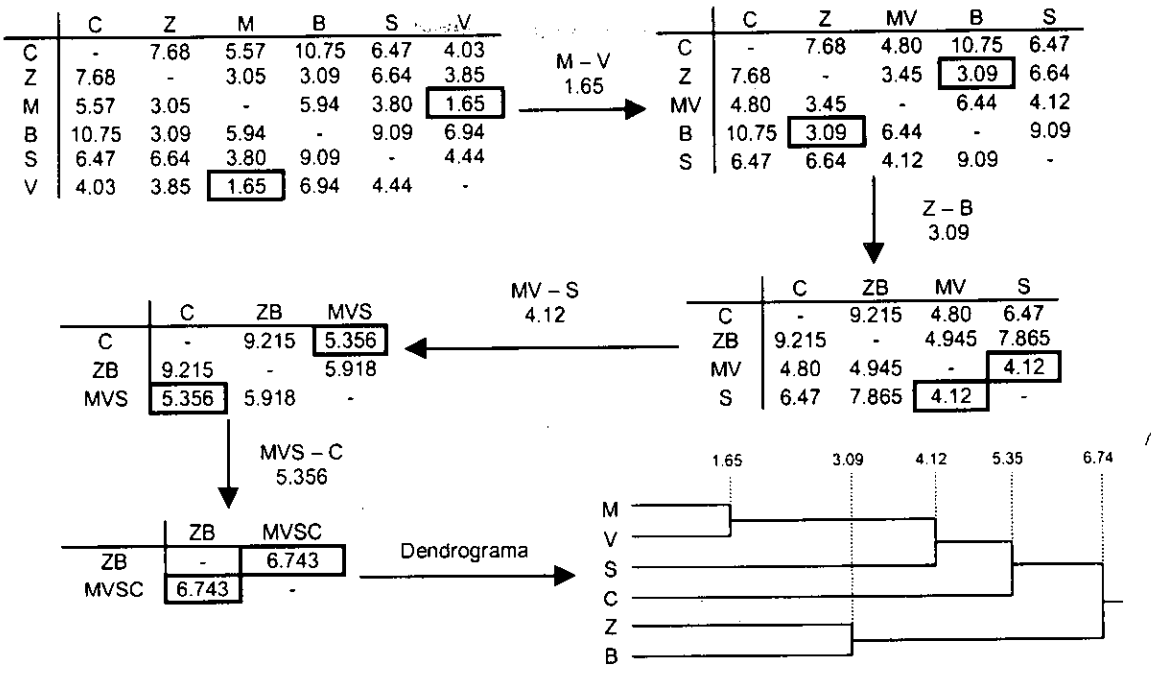


Figura 6.14. Ejemplo de utilización del método del promedio entre grupos utilizando los datos de las ciudades.

Sneath y Sokal (1973) introducen la abreviación UPGMA (Unweighted Pair Group Method using arithmetic Averages) para referirse a esta técnica. La letra U (Unweighted) indica que es un método no ponderado en el que todos los elementos son tratados de la misma forma. En los métodos ponderados (Weighted) los elementos de los clusters más pequeños adquieren más importancia que los elementos de los clusters más grandes. Las siglas PGM (Pair Group Method) indican que los clusters se van uniendo por pares y la letra A (arithmetic Averages) indica que al unir los clusters se halla la media aritmética de las distancias existentes entre sus elementos (otros métodos llevan la letra C indicando que las distancias se miden entre los centroides de cada grupo).

Método de la media (weighted group average - WPGMA -).

El método de la media es similar al del promedio entre grupos pero en este caso no se tiene en cuenta el número de elementos que forman cada cluster. Por esta razón se dice que este es un método ponderado ya que los elementos de los clusters más pequeños adquieren más relevancia que los elementos de los clusters mayores. Se suele conocer con las siglas WPGMA (Weighted Pair Group Method using arithmetic Averages) y se ilustra en la Figura 6.15. En ella se acaba de formar el cluster A uniendo el elemento 3 a los elementos 4 y 5 ya unidos. La distancia de este nuevo cluster con el cluster B formado por los elementos 1 y 2 resulta de hallar la media entre las distancias $d_{12\text{-}3}$ y $d_{12\text{-}45}$.

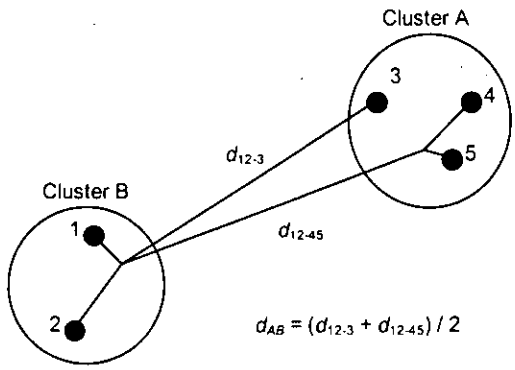


Figura 6.15. Distancia según el método de la media, el cluster A acaba de formarse uniendo el elemento 3 a los elementos 4 y 5 ya unidos.

En este método no se tiene en cuenta el número de elementos que forma cada cluster. Si en el cluster A tuviéramos 50 elementos y en el cluster B sólo uno, el proceso de unión sería idéntico, con lo cual la importancia del elemento del cluster B sería mayor que la que tuvieran cada uno de los 50 elementos del cluster A (los elementos del cluster B están ponderados). También se puede ver que en este caso no es necesario retener las distancias iniciales entre pares de elementos (como ocurría en el método del promedio entre grupos en el que las distancias que aparecían en la matriz inicial debían preservarse a lo largo de todo el proceso).

Aplicando este algoritmo a la matriz de similitudes para los datos de las ciudades obtenemos el resultado de la Figura 6.16.

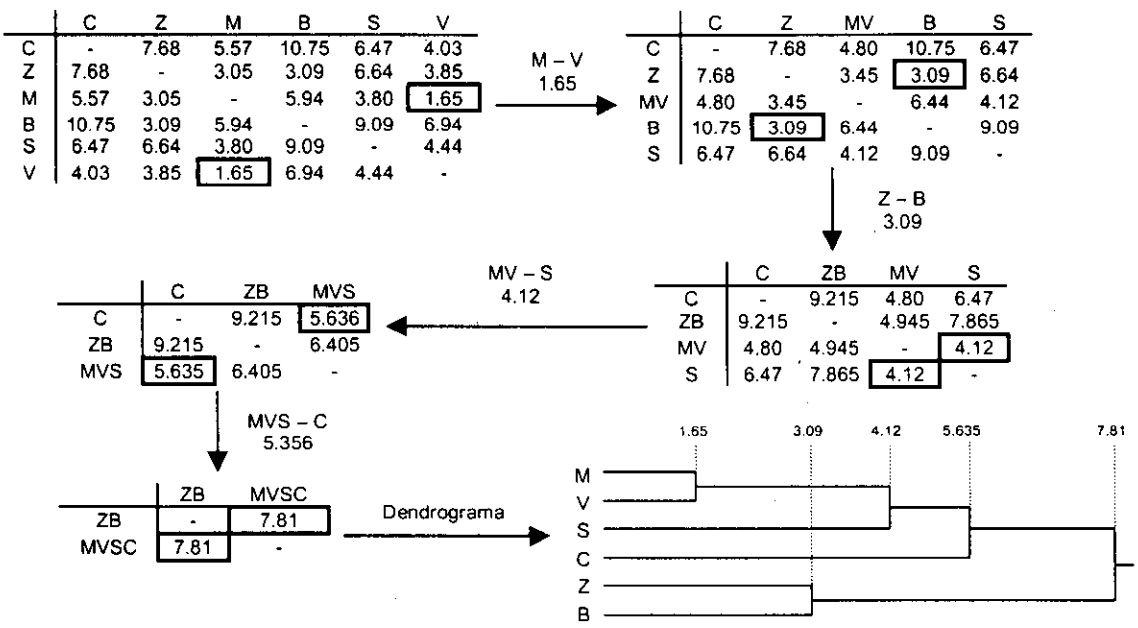


Figura 6.16. Ejemplo de utilización del método de la media utilizando los datos de las ciudades.

Método del centroide (Centroid - UPGMC -)

En este método, una vez que se han formado los clusters, éstos son representados por su centroide (aquel elemento cuyas coordenadas se calculan a partir de la media de las coordenadas de los elementos que forman el cluster). El método se ilustra en la

Figura 6.17. en la que vemos que la distancia entre los clusters A y B se mide como la distancia existente entre sus centroides.

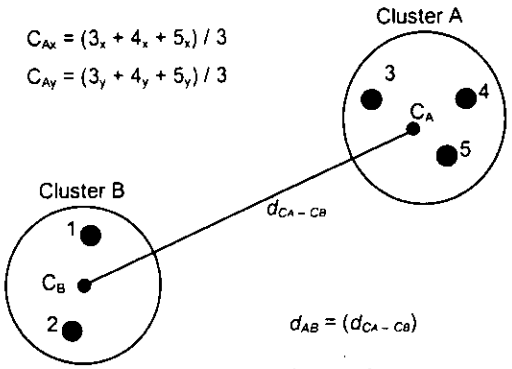


Figura 6.17. Distancia según el método del centroide. La distancia entre A y B es la distancia entre sus centroides. Las coordenadas del centroide se calculan a partir de la media de las coordenadas de los elementos que forman el cluster.

Como vemos este método plantea una variación con respecto a los demás y consiste en que es necesario conocer las coordenadas que definen a cada elemento en el espacio, es decir conocer la matriz de elementos, además de la matriz de distancias entre pares de elementos. Aplicando este método al ejemplo de las ciudades obtenemos los resultados de la Figura 6.18.

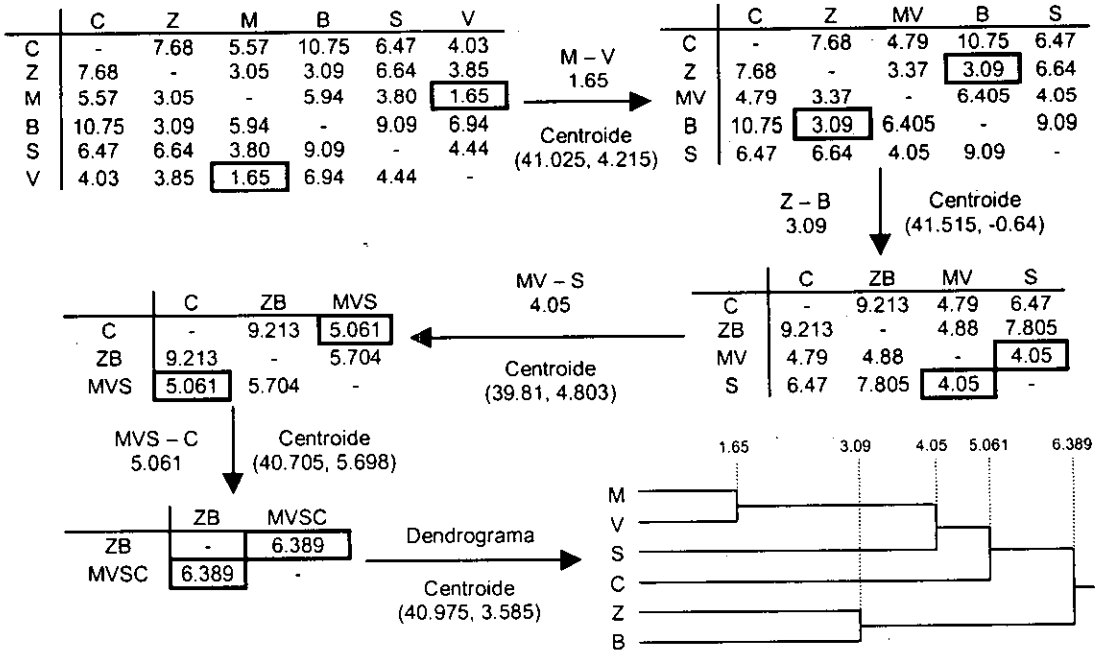


Figura 6.18. Ejemplo de utilización del método del centroide utilizando los datos de las ciudades.

Método de la mediana (Median - WPGMC -)

El método de la mediana también se conoce como el método de Gower ya que fue él quien lo propuso por primera vez (Gower, 1967). Este método es similar al método del centroide pero aquí no se tiene en cuenta el número de elementos que forman los clusters que se van a unir (se trata de un método ponderado). En este caso la distancia entre el cluster B (formado por los elementos 1 y 2) y el cluster A (formado por la unión del elemento 3 con los elementos 4 y 5 ya agrupados) es la mediana del

triángulo formado por el centroide del grupo B, el centroide de 3 y el centroide de 4-5 (Figura 6.19).

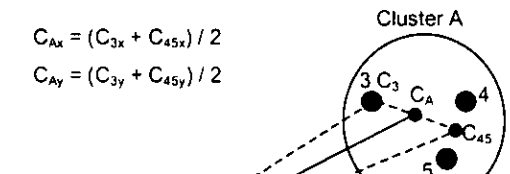


Figura 6.19. Distancia según el método de la mediana. La distancia entre A y B es la distancia entre sus centroides. Las coordenadas del centroide se calculan a partir de la media de las coordenadas de los centroides de los grupos que se unen para formar el nuevo cluster.

Con los métodos ponderados, como el de la mediana o el de la media lo que se pretende es que no se pierdan las características de los grupos más pequeños al unir estos con grupos más grandes. Esto se consigue no considerando los elementos por separado y sólo teniendo en cuenta en cada cluster un dato (su centroide en el caso de la mediana o su distancia a los otros grupos en el caso de la media) sin importar que número de elementos forman ese cluster. De esta forma estamos ponderando en importancia a los elementos de los clusters más pequeños con los elementos de los clusters más grandes. Aplicando este método a los datos de las ciudades obtenemos los resultados que se muestran en la Figura 6.20.

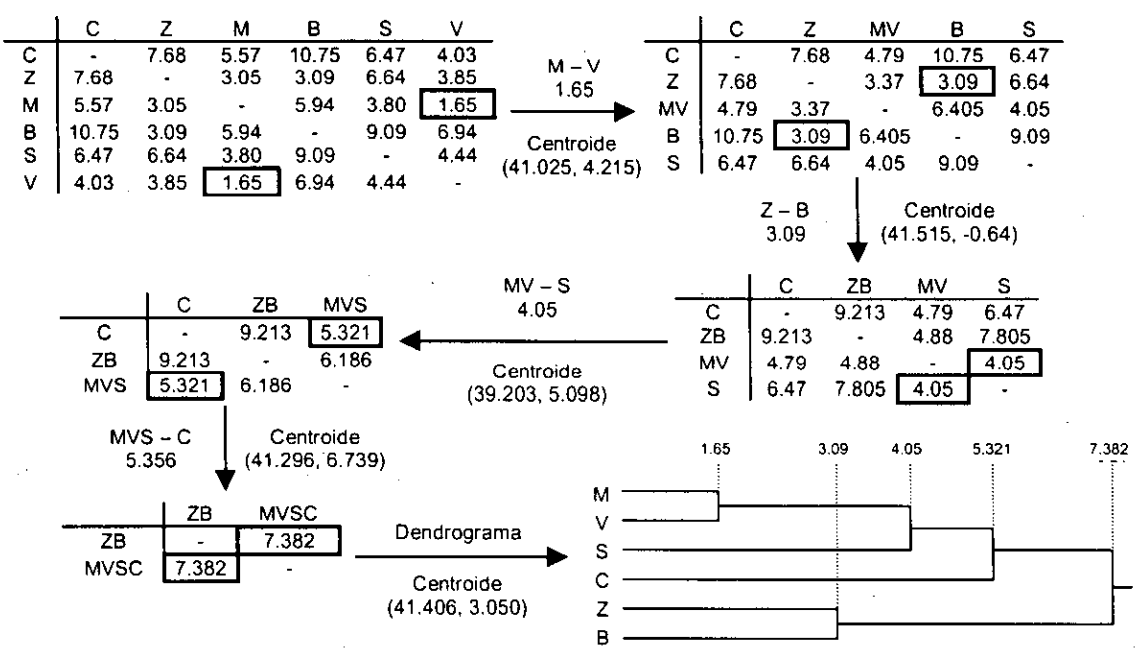


Figura 6.20. Ejemplo de utilización del método de la mediana utilizando los datos de las ciudades.

Método de Ward

El método de Ward también se denomina el método del *mínimo error cuadrático* ya que une, en cada nivel, los dos clusters que minimizan el error cuadrático entre todas las posibles uniones de pares de clusters existentes. Hay que tener en cuenta, sin embargo, que el método de Ward no obtiene la partición que minimiza el error cuadrático de entre todas las posibles particiones con ese número de clusters.

La idea que subyacen en el método de Ward puede verse de forma fácil si consideramos datos univariantes (Everitt, 1993). Supongamos, por ejemplo, 10 individuos con los siguientes valores (2, 6, 5, 6, 2, 2, 2, 0, 0, 0) de una variable dada. La perdida de información que se produce al tratar los diez elementos como un grupo con media 2.5 se representa mediante el error de suma de cuadrados (*ESS*).

$$ESS = \sum_{i=1}^{n} (x_i - \bar{x})^2$$

equ. 6.47

Para este caso ESS = 50.5. Si los diez individuos se agrupan en los siguientes cuatro clusters {0, 0, 0}, {2, 2, 2, 2}, {5}, {6, 6} el *ESS* puede calcularse como la suma de los *ESS* de cada grupo lo que nos da que

$$ESS_{total} = ESS_{grupo\ 1} + ESS_{grupo\ 2} + ESS_{grupo\ 3} + ESS_{grupo\ 4} = 0.0$$

Para el caso de que los datos sean multivariables debe extenderse la equ. 6.47 para abarcar todas las p posibles dimensiones

$$ESS = \sum_{i=1}^{n} \sum_{j=1}^{p} (x_{ij} - \bar{x}_j)^2$$

equ. 6.48

Cuando en un clustering jerárquico se unen los clusters i y j para formar el cluster k el incremento del error total de suma de cuadrados producido es

$$\Delta ESS_{total} = ESS_k - ESS_i - ESS_j$$

equ. 6.49

En este método hay que tener en cuenta que la escala del dendrograma que se obtiene no es la misma que la escala utilizada para medir la distancia entre elementos. Así las jerarquías se pueden unir a niveles varios cientos de veces mayores que las distancias existentes entre pares de elementos. El efecto visual que se produce es que se acentúa el *tiempo de vida* de los clusters.

Al igual que el método del centroide y el método de la mediana, el método de Ward trabaja con la matriz de elementos inicial y no con la matriz de similitudes, por lo que no se considera adecuado para la validación de sistemas expertos tal y como venimos haciendo hasta ahora.

La ecuación de Lance y Williams

Lance y Williams (1967) obtuvieron una ecuación que resume todos los métodos vistos para obtener la distancia entre dos clusters. Según esta ecuación la distancia entre el cluster k y el cluster (ij), formado de la fusión de los clusters i y j, es:

$$d_{k(ij)} = \alpha_i d_{ki} + \alpha_j d_{kj} + \beta d_{ij} + \gamma |d_{ki} - d_{kj}|$$

equ. 6.50

en donde d_{ij} es la distancia entre los clusters i y j. De esta forma sólo hay que variar los valores de los parámetros α, β, y γ para obtener los distintos métodos de clustering según se ve en la Tabla 6.26 (en la que n_i representa el número de individuos del cluster i).

Método de Clustering	α_i	α_j	β	γ
distancias mínimas	½	½	0	-½
distancias máximas	½	½	0	½
promedio entregrupos (UPGMA)	$\frac{n_i}{n_i + n_j}$	$\frac{n_j}{n_i + n_j}$	0	0
media (WPGMA)	½	½	0	0
centroide (UPGMC)	$\frac{n_i}{n_i + n_j}$	$\frac{n_j}{n_i + n_j}$	$\frac{-n_i n_j}{(n_i + n_j)^2}$	0
mediana (WPGMC)	½	½	-¼	0
Ward	$\frac{n_k + n_i}{n_k + n_i + n_j}$	$\frac{n_k + n_j}{n_k + n_i + n_j}$	$\frac{-n_k}{n_k + n_i + n_j}$	0
Flexible	α	α	$1-2\alpha$	0

Tabla 6.26. Coeficientes para la ecuación de Lance y Williams.

Al describir el método del centroide, el de la mediana y el de Ward habíamos comentado que era necesario conocer las coordenadas que definen a los elementos para poder aplicarlos; sin embargo, con la ecuación de Lance y Williams podemos trabajar directamente con la matriz de distancias, siempre y cuando se utilice la distancia euclídea al cuadrado. Es posible utilizar otro tipo de distancias pero esto puede conducir a resultados extraños de difícil interpretación (Anderberg, 1973).

El último método que aparece en la tabla es el método flexible, sugerido por Lance y Williams (1967), y que no especifica un valor concreto de los parámetros sino unas restricciones que deben cumplir los mismos. Estas restricciones son $\alpha_i + \alpha_j + \beta = 1$, $\alpha_i = \alpha_j$, $\beta < 1$ y $\gamma = 0$; lo que implica que los parámetros finales queden como $\alpha_i = \alpha$, $\alpha_j = \alpha$, $\beta = 1-2\alpha$, y $\gamma = 0$.

Si permitimos que β varíe podemos obtener modelos de clustering que presenten diversas características. Lance y Williams descubrieron que los más útiles eran valores negativos y pequeños de β, y en sus ejemplos utilizaron $\beta = -¼$ aunque otros autores han sugerido también la utilización de $\beta = -½$ (Scheibler y Schneider, 1985).

Comparación entre los distintos métodos

Muchas de las características que definen a los métodos de clustering jerárquico y aglomerativo ya han sido descritas en los apartados anteriores, sin embargo, en este apartado se pretende analizar una comparación de los distintos métodos para intentar determinar cual sería el más adecuado para cada caso.

Uno de los defectos que más se le achaca al método de las distancias mínimas es el conocido con el nombre de *encadenamiento* (chaining). Éste efecto consiste en que el método de las distancias mínimas tiende a unir juntos clusters que, aunque son claramente distintos, están unidos por una serie de elementos intermedios. Este efecto puede verse claramente en la Figura 6.21. En ella tenemos la gráfica (a) en la que vemos dos nubes de puntos claramente diferenciadas que están unidas por una línea de puntos intermedios. Si realizamos un análisis cluster mediante el método de las distancias mínimas e indicamos que queremos obtener dos clusters obtenemos la figura (b) en la que un cluster es el punto blanco en la parte superior de la gráfica y el resto de puntos es el otro cluster. Este resultado es claramente opuesto al que se vislumbra en la gráfica (a). Los puntos intermedios entre las dos nubes de puntos se denominan *puntos de ruido* (Everitt, 1993).

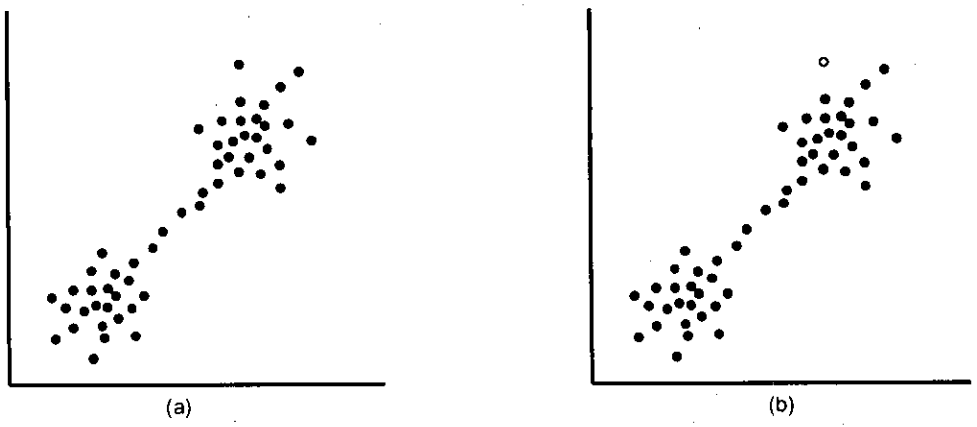


Figura 6.21. Efecto de encadenamiento en el método de las distancias mínimas. (a) dos nubes de puntos claramente diferenciadas unidas por una línea de puntos intermedios. (b) dos clusters obtenidos por el método de las distancias mínimas (uno es el punto blanco en la parte superior de la gráfica y el otro es el resto de puntos).

Sin embargo, autores como Jardine y Sibson (1968) no consideran al encadenamiento como un defecto e indican que el método de las distancias mínimas puede llegar a ser más eficiente que otros. En trabajos posteriores, Jardine y Sibson (1971) basan sus preferencias por el método de las distancias mínimas basándose en que es el único método que cumple una serie de condiciones matemáticas (continuidad, mínima distorsión, etc.)

Si el método de las distancias mínimas da como resultados clusters de forma más bien alargada en los que elementos bastante distintos pueden llegar a unirse en el mismo cluster debido al encadenamiento, el método de las distancias máximas produce el

efecto inverso, se obtienen muchos clusters con una distancia interior muy pequeña. Ahora lo que puede ocurrir es que elementos bastante similares tarden mucho en unirse en el mismo cluster (*efecto disección*). Por estos motivos se dice que el método de las distancias mínimas es *espacio contractivo,* mientras que el método de las distancias máximas es *espacio dilatante*. La necesidad de un compromiso entre estos dos extremos ha motivado la aparición de técnicas como el promedio entre grupos, y el resto de métodos vistos en los apartados anteriores, que se caracterizan por ser *espacio conservativos* (Kaufman y Rousseeuw, 1990). Autores como Jobson (1992) señalan que el método del centroide es espacio contractivo y el de Ward espacio dilatante. El método flexible depende de los valores que se le den a sus parámetros para determinar si es conservativo, contractivo o dilatante (Cuadras, 1991).

La ventaja que presenta el método de las distancias mínimas y el método de las distancias máximas es que las uniones de clusters se producen en distancias que están presentes en la matriz inicial de distancia. Ningún otro método de clustering tiene estas características y esto permite realizar algoritmos de clustering más eficaces y que pueden ser aplicados a un conjunto de datos mayor. Además estos métodos se caracterizan por ser invariables ante transformaciones monótonas de la matriz de distancias. Esto significa que los resultados obtenidos serán los mismos si variamos los datos de la matriz de distancias pero preservamos el mismo orden de rangos que en la matriz original. De esta forma las dificultades que implica la combinación y el escalado de diferentes variables en una medida de proximidad son de menor importancia, y suele ser muy útil cuando utilizamos disimilitudes ordinales.

Otra ventaja que tiene el método del las distancias mínimas es su forma de proceder en presencia de ligaduras (aparición de dos distancias idénticas en la matriz de distancias). En los métodos vistos hasta ahora, excepto en el de las distancias mínimas, la decisión de tomar una u otra de las distancias ligadas puede provocar la aparición de dendrogramas completamente distintos. Sin embargo, Jardine y Sibson (1971) demostraron que el método de las distancias mínimas tenía la propiedad de *continuidad*, por la cual los distintos clusters se iban uniendo de forma suave a medida que se acercaban unos a otros, sin importar el orden tomado en las ligaduras. En la Figura 6.22 se puede ver claramente este efecto, partiendo de la matriz de distancias dada vemos que en el primer paso existe un empate entre las distancias d_{AB} y d_{BC}. Utilizando el método de las distancias mínimas el dendrograma obtenido es el mismo tanto si unimos primero A y B como si unimos B y C. Sin embargo utilizando, por ejemplo, el método de la media, los dendrogramas varían drásticamente dependiendo si empezamos por unir A-B o B-C.

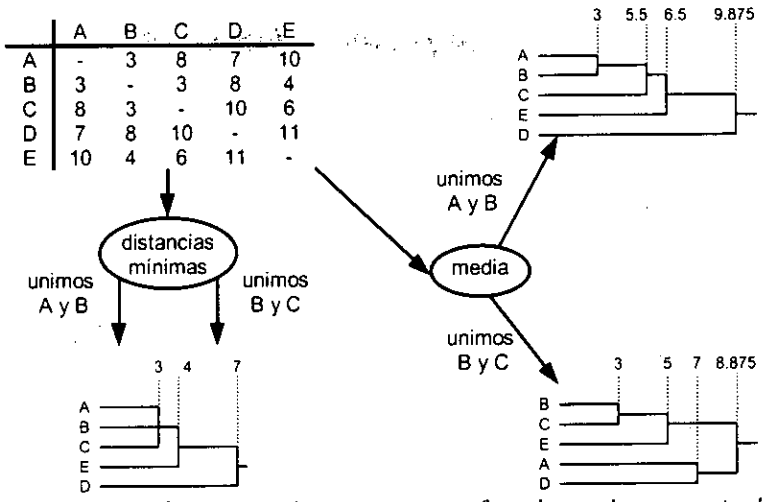


Figura 6.22. El método de las distancias mínimas no se ve afectado por la presencia de ligaduras en la matriz de distancias mientras que los otros métodos (por ejemplo el de la media) pueden presentar resultados finales muy diferentes.

Métodos como el del promedio entre grupos y el de la media son muy populares, ya que obtienen información de todos los elementos que forman el cluster y no sólo de los más cercanos o los más alejados. En el método del promedio entre grupos se tiene en cuenta el tamaño de los clusters a la hora de realizar las uniones, por lo que los nuevos clusters formados siempre estarán más próximos a los clusters con muchos elementos. En el método de la media el tamaño de los clusters no se tienen en consideración al realizar la unión. Esto tiene la ventaja de que no se pierden las características de los clusters más pequeños, pero presenta el inconveniente de que los elementos incluidos en clusters pequeños sean más representativos que los elementos incluidos en los clusters más grandes.

Los métodos del centroide y de la mediana son equivalentes a los métodos del promedio entre grupos y de la media, sólo que en este caso se trabaja con los centroides de los clusters. Estos métodos se caracterizan por tener una interpretación geométrica directa mientras que el promedio entre grupos y la media no presentan una interpretación geométrica sencilla (Dubes, 1993). El principal problema que plantean el método del centroide y el de la mediana es que necesitan conocer la matriz de elementos además de la matriz de distancias. En la ecuación de Lance y Williams no es necesario el conocimiento de la matriz de elementos pero la distancia utilizada se limita a la distancia euclídea al cuadrado (la misma situación aparece con el método de Ward).

Otro problema que presentan el método del centroide y el método de la mediana es que no son monótonos. Decimos que un método de clustering es *monótono* si, cuando unimos los cluster i y j en el cluster ij, entonces para todos los clusters k distintos de i y j se cumple que $d_{k\text{-}ij} \geq d_{ij}$. La monotoneidad permite dibujar a los dendrogramas como árboles binarios y permite que se satisfaga la distancia cofenética. Sin la monotoneidad pueden aparecer cruces y vueltas atrás cuando dos clusters se unen a niveles inferiores a los de los propios clusters antes de unirse. Así se deduce que interpretar un dendrograma no monótono es extremadamente difícil. Para que un método de clustering sea monótono tiene que cumplir que $\alpha_i + \alpha_j + \beta \geq 1$ (Cuadras, 1991).

La diferencia entre el método del centroide y el de la mediana es similar a la existente entre el método del promedio entre grupos y el de la media, y se puede ver claramente en la Figura 6.23. En el método del centroide los grupos con muchos

elementos absorben a los grupos pequeños, que quedan prácticamente eliminados. Por otro lado el método de la mediana permite que permanezcan las características de los grupos pequeños a costa de darle más importancia a los individuos que los forman.

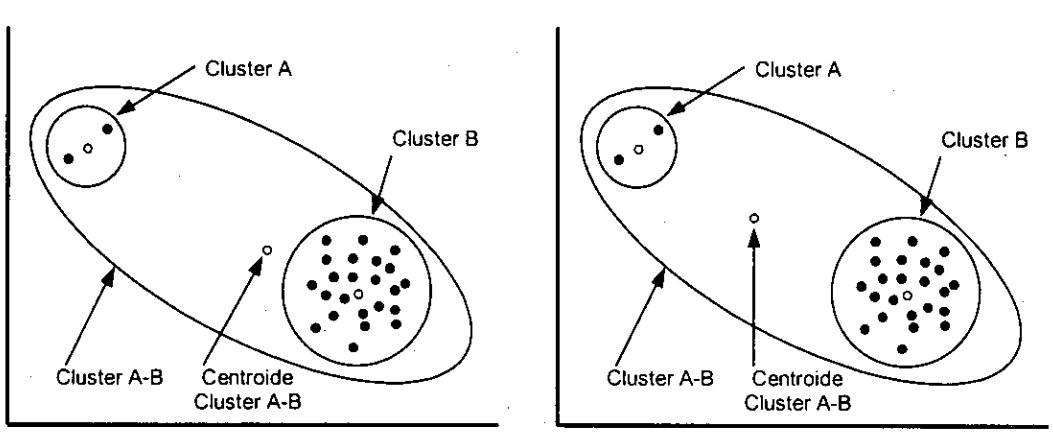


Figura 6.23. Comparación entre el método del centroide (a) y el método de la mediana (b). El método del centroide da la misma importancia a todos los puntos por lo que el centroide del nuevo cluster estará más cerca del cluster mayor. El método no tiene en cuenta el número de elementos de cada cluster y el nuevo centroide se halla a la misma distancia de los clusters originales.

Por último podemos decir que, a pesar de los múltiples estudios que se han llevado a cabo, no existe ningún método que sea claramente superior a los demás para cualquier tipo de datos (Milligan, 1980). De todas formas el método que generalmente suele ser sugerido por todos los autores es el del promedio entre grupos. Kaufman y Rousseeuw (1990) lo recomiendan porque es el único que cumple tres condiciones que consideran fundamentales:

(1) **Las disimilaridades entre los clusters que se unen son monótonas**, lo que permite representarlo mediante un dendrograma. Los únicos métodos no monótonos son el del centroide y el de la mediana.

(2) **Las disimilaridades entre los clusters no son ambiguas**, lo que significa que la distancia entre dos clusters siempre será la misma independientemente del orden en que se hayan unido previamente los elementos que los componen. Los métodos que cumplen esta propiedad son el de las distancias mínimas, el de las distancias máximas y el del promedio entre grupos (ya que las distancias entre dos clusters se definen explícitamente en términos de las distancias originales existentes entre los objetos). Lo mismo puede decirse del método del centroide y del método de Ward cuando pueden ser definidos por puntos en un espacio euclídeo porque lo único que tenemos que hacer es calcular la distancia euclídea entre los centroides de los clusters.

(3) **Las disimilaridades entre los clusters son estadísticamente consistentes**, significa que las disimilaridades entre los clusters no deben variar demasiado si consideramos más elementos en nuestra muestra. Así vemos que podemos descartar métodos como el de Ward (en el que los niveles de agrupamiento tienden a crecer mucho más que las disimilaridades entre los elementos), el de las distancias mínimas (en el que las distancias tienden a

cero) y el de las distancias máximas (en el que las distancias tienden al infinito). Los únicos métodos que cumplen esta propiedad son el método del centroide y el método del promedio entre grupos. Aunque esta condición pueda considerarse un poco restrictiva, Kaufman y Rousseeuw no creen que sea así ya que debería ser natural esperar que al añadir más puntos el clustering nos condujera a resultados más correctos.

En base a la discusión desarrollada sobre los distintos algoritmos de análisis cluster podemos decir que, aunque no haya un método claramente superior a otro, si hay métodos (como el promedio entre grupos) que tienen un comportamiento más adecuado en la mayoría de las ocasiones.

6.2.2.7. Métodos de representación

Existen muchas formas de representar gráficamente los resultados de un análisis cluster jerárquico. La más comúnmente utilizada, y la que mayor impacto visual produce es el dendrograma que ya hemos visto varias veces a lo largo de este trabajo. Las ventajas del dendrograma son evidentes, resulta muy fácil hacerse una idea de cómo se ha realizado la unión de los diversos grupos y en que niveles se han producido estas uniones. El único problema que plantean es que si el número de elementos es demasiado elevado (alrededor de 200) son difíciles de interpretar, además necesitan un cierto nivel de complejidad gráfica para representarse correctamente (y evitar que existan líneas de cruces que perturben su análisis).

Otro tipo de representaciones comúnmente utilizadas son los carámbanos o témpanos (icicles) que se representan en la Figura 6.24 junto con el dendrograma correspondiente al mismo análisis cluster. Los carámbanos muestran el proceso de unión de los grupos mediante una serie de barras verticales que se parecen caer del techo como carámbanos helados (de ahí su nombre).

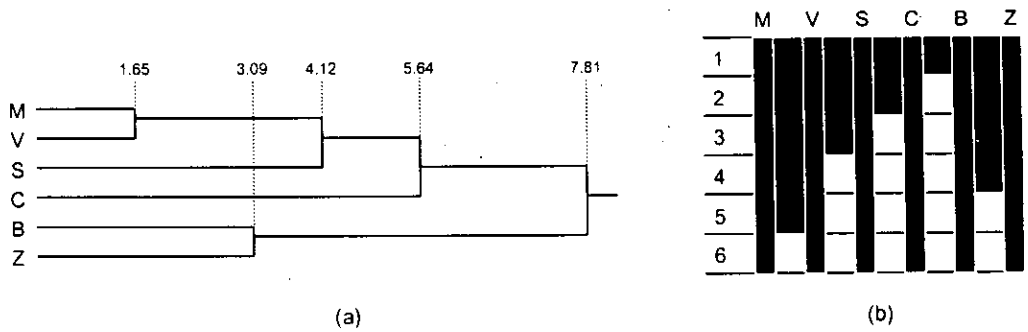


Figura 6.24. (a) dendrograma del análisis cluster para el método de la media con los datos de ciudades. (b) gráfica de carámbanos correspondiente.

En la Figura 6.24, las variables se representan en columnas y los niveles de agregación en filas. El nivel número 1 corresponde al último paso del análisis, es decir, cuando todos los clusters se encuentran unidos en uno solo. De la misma forma el último nivel de agregación corresponde al primer paso del análisis, en el que cada elemento forma un cluster. Los clusters se representan mediante una serie de barras verticales separadas por un espacio en blanco. Cuando dos clusters se unen el espacio en blanco que había entre ellos desaparece.

Lo que se pretende con este gráfico es encontrar de forma rápida soluciones con un número determinado de clusters (para encontrar una solución con 3 clusters solo hay que ir al nivel 3 y observar cuales son los elementos que están unidos, en la gráfica de la Figura 6.24 serían MVS-C-BZ). Así vemos que los dendrogramas tratan de resaltar las distancias entre los distintos clusters, mientras que los carámbanos tratan de facilitar la búsqueda de un determinado número de clusters. Una ventaja que tienen los carámbanos con respecto al dendrograma es que no hace falta capacidades gráficas especialmente complicadas para poder representarlos.

También, si queremos mezclar las características de ambos métodos podemos incluir los valores de los niveles de agregación en la gráfica de carámbanos y los índices de agregación en los dendrogramas. Otras variaciones que pueden aparecer es colocar los dendrogramas de forma vertical o los carámbanos en sentido horizontal, sin embargo la forma común en la que suelen aparecer en la bibliografía es la que aquí se ha representado.

Un tipo de gráfico muy intuitivo y de sencilla interpretación es el *gráfico de burbujas* (Kaufman y Rousseeuw, 1990), también conocido como *gráfico de curvas* o *loop plot*. Este tipo de representación esta especialmente destinada a elementos que puedan representarse en gráficas de dos dimensiones (Figura 6.25).

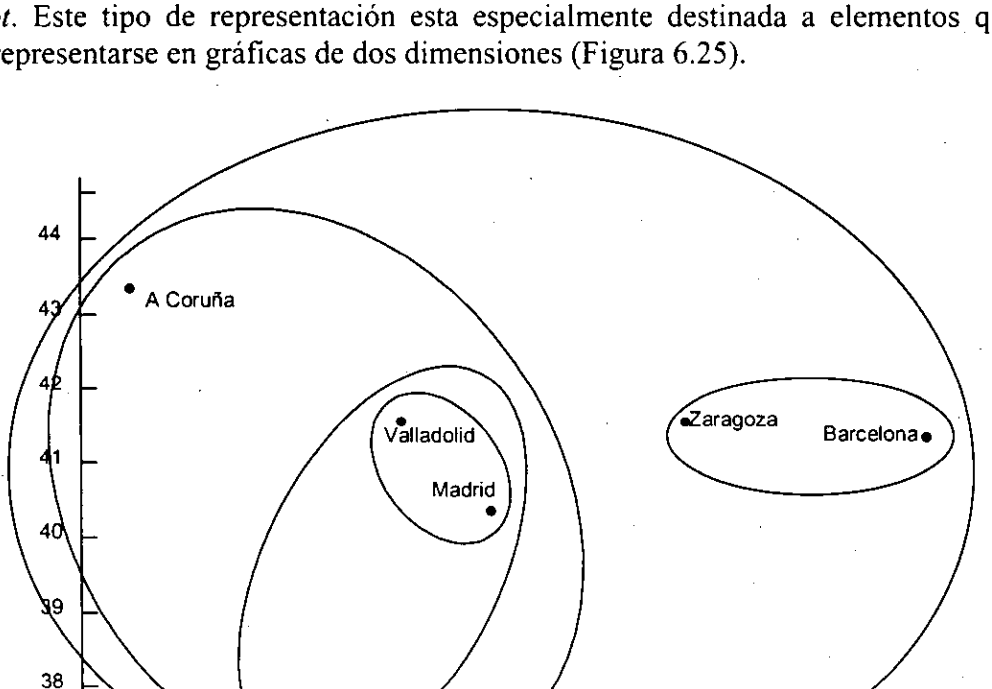


Figura 6.25. Representación de seis ciudades según sus coordenadas x e y con las burbujas correspondientes al análisis cluster.

Cada vez que un cluster se une con otro se forma una burbuja que los rodea. Realizando este proceso para cada uno de los niveles de agregación se obtiene una representación de fácil interpretación que produce un elevado impacto visual. El principal inconveniente de esta técnica es que está limitada a aquellos elementos que se puedan representar en 2D, además de ser bastante complicada ya que exige que las

curvas se tracen de forma adecuada para evitar cruces que dificultarían la comprensión de la representación.

Otros tipos de representación del análisis cluster no emplean gráficos sino tablas de aglomeración (Bisquerra, 1989). Estas tablas pretenden ser un resumen en modo texto del resultado del análisis cluster y, generalmente, indican qué clusters se unen en cada paso y los índices o coeficientes de agregación. El modelo más sencillo es el que se presenta en la Tabla 6.27, en el que aparecen 3 columnas (dos para representar los clusters unidos y una para indicar el coeficiente de agregación), y tantas filas como pasos hemos realizado para unir todos los clusters en uno solo.

Cluster 1	Cluster 2	Coeficiente
M	V	1.65
B	Z	3.09
MV	S	4.12
MVS	C	5.64
MVSC	BZ	7.81

Tabla 6.27. Tabla de agregación del análisis cluster.

Este tipo de tablas son muy intuitivas, sin embargo presentan el defecto de que si el número de elementos es muy grande las celdas deben ser también de tamaño considerable para almacenar todos los elementos. Por ello otros paquetes de software como el SPSS (Norusis, 1995) muestran unas tablas de agregación más complicadas de interpretar pero que no presentan estos problemas (Tabla 6.28).

Etapa	Clusters combinados		Coeficiente	Etapa en la que primero aparecen		Sig. etapa
	Cluster 1	Cluster 2		Cluster 1	Cluster 2	
1	M	V	1.65	0	0	3
2	Z	B	3.09	0	0	5
3	M	S	4.12	1	0	4
4	C	M	5.64	0	3	5
5	C	Z	7.81	4	2	0

Tabla 6.28. Tabla de agregación del análisis cluster (SPSS).

En esta tabla cada fila representa los distintos pasos de agregación del análisis cluster. La columna *Etapa* indica en que paso no hallamos, las columnas *Cluster 1* y *Cluster 2* que se hallan bajo el título de *Clusters combinados* indican qué clusters se han combinado en ese paso (los nombres pueden representar un cluster con un único elemento o un cluster con varios elementos). *Coeficiente* indica el nivel en el que se han unido los clusters indicados. Las columnas bajo el título *Etapa en la que primero aparecen* indican en que etapa se ha formado un cluster que tiene varios elementos y la columna *Sig. etapa* indica la siguiente etapa a la cual otro cluster es unido con el que se ha formado en este paso.

6.2.2.8. Validación del análisis cluster

Las técnicas de clustering se emplean para descubrir estructuras en los datos, sin embargo, como indican Aldenderfer y Blashfield (1984), su modo de operar es de "imposición de estructura". Es decir, los algoritmos siempre encontrarán una estructura en grupos aunque esta realmente no exista. Por ello es importante dilucidar si los grupos descubiertos por el análisis son reales o son impuestos por el método utilizado.

Centrándonos en lo que es el análisis cluster jerárquico, sabemos que dicho análisis produce una secuencia de particiones anidadas, que parten de n clusters con un

solo elemento a un único cluster que contiene los n elementos. El uso que se le dé a esta secuencia de particiones depende de la aplicación en particular que estemos considerando. Así, en algunos casos la jerarquía indexada puede utilizarse para resumir las relaciones entre varios subgrupos (un ejemplo serían los grupos de animales o plantas en una taxonomía numérica). Por otro lado podemos desear tan sólo una posible partición de la jerarquía para obtener, por ejemplo, el número de clusters deseados para un posterior análisis. Estos aspectos requieren un análisis más detallado que la simple obtención del dendrograma del análisis cluster.

Correlación cofenética

El método más comúnmente usado para evaluar el acuerdo existente entre el dendrograma obtenido tras el análisis cluster y la matriz de distancias inicial es el *coeficiente de correlación cofenética*. Este coeficiente no es más que el grado de correlación de Pearson existente entre la matriz de distancias inicial y la matriz de distancias cofenéticas. La magnitud de este índice debe estar cercana a la unidad si la solución del análisis cluster es buena.

También se puede destacar que dado que en el dendrograma sólo existen (n-1) valores de proximidad y la matriz de distancias cofenéticas tiene $n(n\text{-}1)/2$ entradas, muchos de estos valores serán similares. Un ejemplo de la correlación cofenética se puede ver en la Figura 6.26.

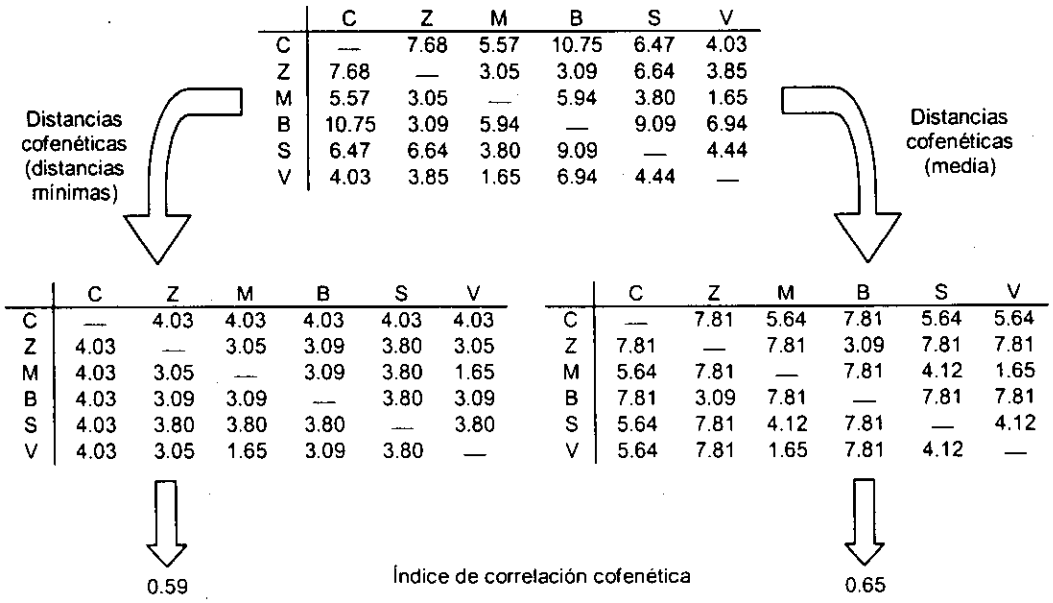


Figura 6.26. Índice de correlación cofenética para los datos del porcentaje de acuerdo utilizando el método de las distancias mínimas y el método de la media.

Stress

Un método alternativo para comparar dos conjuntos de proximidades en el caso de utilizar distancias euclídeas es calcular la medida de stress

$$\frac{\sum_{\substack{i=1 \\ i<j}}^{n}\sum_{j=1}^{n}(p_{ij}-\hat{p}_{ij})^2}{\sum_{\substack{i=1 \\ i<j}}^{n}\sum_{j=1}^{n}\hat{p}_{ij}^2}$$

equ. 6.51

en donde p_{ij} denota la distancia original y $\hat{p}_{ij}$ la distancia derivada. Esta medida se usa normalmente en análisis de escalamiento multidimensional para evaluar las soluciones obtenidas.

Distancias basadas en centroides

Otra solución alternativa para derivar las distancias de los resultados de un análisis cluster jerárquico sería volver a cada uno de los pasos, y computar la distancia existente entre los centroides de los clusters que se unen a cada paso. En este caso el método del centroide no se utiliza como criterio para formar los clusters, sino como medida de la proximidad entre los mismos, en este caso, sin embargo, las distancias derivadas pueden no satisfacer la desigualdad ultramétrica.

Elección del número de clusters

En el análisis cluster jerárquico muchas veces no sólo se quiere determinar cual es el mejor método sino también ver cuál sería la mejor partición del dendrograma, es decir, decidir el número de clusters que resultan de nuestro análisis. En el análisis cluster jerárquico se suele utilizar un método informal pero que normalmente suele dar buenos resultados. Este método consiste en analizar los niveles de fusión de los distintos clusters, seleccionar el paso en el que se produce la mayor diferencia entre los niveles de fusión y considerar que el número de clusters ideal era el existente antes de dar dicho paso. Un ejemplo muy claro lo vemos en la Figura 6.27, en ella vemos como el salto mayor se produce cuando fusionamos los dos últimos clusters, lo que claramente sugiere que la solución ideal es la de dos clusters.

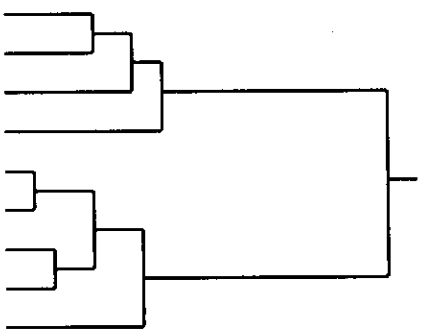


Figura 6.27. Dendrograma indicando dos grupos.

Para los datos de las ciudades agrupados mediante el método de la media obtenemos el siguiente resultado (Figura 6.28).

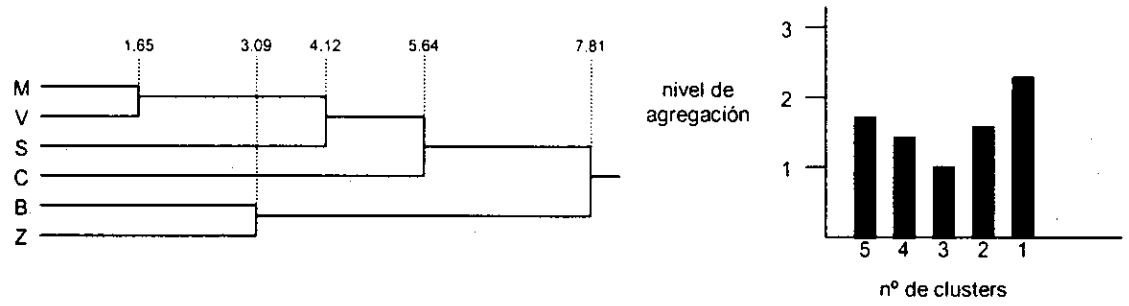


Figura 6.28. Las variaciones producidas en los niveles de agregación determinan la existencia de dos clusters.

En Figura 6.28 vemos como la mayor variación en el nivel de agregación se produce cuando juntamos todos los clusters en uno solo. Esto indica que la solución ideal sería la anterior, es decir, la configuración con dos clusters MVSC y BZ.

6.2.3. Escalamiento Multidimensional

En la Figura 6.25 veíamos como una serie de ciudades eran dibujadas en una gráfica *X-Y* a partir de sus coordenadas. A partir de esta gráfica es fácil establecer un matriz en la que se representa, para cada par de elementos, la similitud o disimilitud existente entre ellos y que representabamos en la Tabla 6.25.

El problema que nos planteamos a continuación es el siguiente: partiendo de la matriz de (di)similitudes ¿es posible obtener una representación gráfica en dos dimensiones de los distintos elementos?. El método para resolver este problema se denomina *Escalamiento Mulditimensional* o *MDS* (Multi-Dimensional Scaling).

Podemos definir el MDS como un conjunto de técnicas de análisis de datos que muestran la estructura de dichos datos como un gráfico geométrico, basándonos en sus similitudes o disimilitudes. El MDS tiene sus orígenes en la psicométrica, en donde fue propuesto para ayudar a comprender los juicios que hacían las personas sobre la similitud existente entre determinados objetos. Torgerson (1958) fue quién propuso el primer método de MDS y quién acuño su término. Los trabajos de Torgerson se basan en investigaciones anteriores llevadas a cabo por Richardson (1938). Con el paso del tiempo el MDS se ha convertido en una técnica de análisis de datos muy utilizada en diversos campos como el marketing, la sociología, la física, la política, la biología, etc.

En nuestro caso utilizaremos el MDS para, a partir de una matriz de similitud que relacione los datos de varios expertos, representar a los expertos como puntos en un espacio bidimensional de tal forma que, puntos cercanos representen similitudes altas y puntos lejanos similitudes bajas.

6.2.3.1. Tipos de escalamiento multidimensional

El MDS toma como origen de su análisis a la matriz de (di)similitudes existentes entre los distintos elementos. Por ello, los diferentes tipos de matrices que podemos encontrar nos van a determinar distintos tipos de MDS.

MDS métrico vs. MDS no métrico

Una primera división que podemos establecer dentro del MDS es la diferenciación entre el MDS métrico y el MDS no métrico. El MDS que está basado en proximidades basadas en medidas directas se conoce como *MDS métrico*, en el que las medidas de similitud se representan en una escala intervalo o ratio. El *MDS no métrico* está basado en proximidades que representan juicios de valor y que se representan en escalas nominales u ordinales. En el MDS métrico la representación espacial intenta preservar las distancias entre los objetos, mientras que en el MDS no métrico la representación espacial solo preserva el orden entre las similitudes.

El ejemplo más típico de MDS métrico es el modelo de Torgerson (1958) mientras que el ejemplo más típico de MDS no métrico es el modelo de Shepard-Kruskal (Shepard, 1962) y (Kruskal, 1964a, 1964b).

MDS de dos vías y un modo vs. MDS de tres vías y dos modos

En una matriz de (di)similitudes podemos distinguir:

1. *vías*: es el número de dimensiones de la matriz.
2. *modos*: es el termino utilizado por Carroll y Arabie (1980) para designar el número de clases de objetos representados en la matriz.
3. *(di)similitudes*: medidas encargadas de medir la similitud o la disimilitud entre los objetos que forman la matriz.

Los métodos citados en el apartado anterior se refieren a matrices de dos vías y una solo modo. Sin embargo, podemos tener matrices de tres vías y dos modos que representen distintas evaluaciones de similitud entre objetos realizadas por varios evaluadores.

El MDS de tres vías y dos modos se suele denominar MDS de diferencias individuales. Este tipo de MDS fue inicialmente esbozado por Tucker y Messick (1963) y posteriormente desarrollado por Carroll y Chang (1970), que desarrollaron el modelo conocido como INDSCAL (INDividual differences SCALing).

6.2.3.2. MDS métrico

Las matrices utilizadas para comparar los resultados de los distintos expertos sobre un determinado test de pares son matrices de dos vías y un modo. En ellas, las similitudes se representan a partir de una medida en la escala intervalo o ratio. Por ello, el tipo de análisis MDS más indicado para nuestra validación es el MDS métrico que describiremos a continuación.

Para poder desarrollar un escalamiento multidimensional métrico en primer lugar hay que disponer de una matriz **D** de orden $(n \times n)$ que asigne una medida de disimilitud $(\delta_{rs}, r, s = 1, 2, \ldots, n)$ a todos los posibles pares de n objetos. Los elementos diagonales de **D** son por lo tanto cero (Figura 6.29).

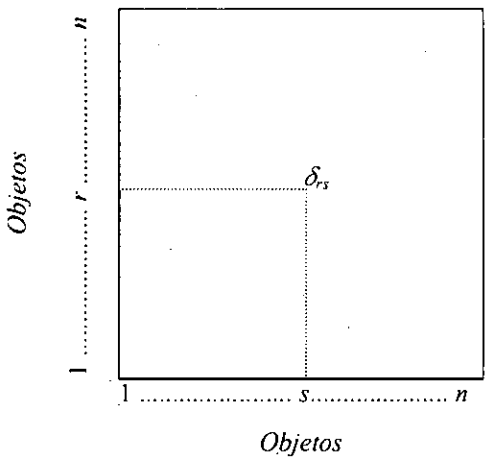


Figura 6.29. Entrada al procedimiento de MDS métrico.

El objetivo del MDS métrico es construir una matriz ($n \times p$), representada en la Figura 6.30, en donde p son un conjunto de dimensiones subyacentes de forma que:

1. Las coordenadas de los n objetos a lo largo de las p dimensiones derivadas permiten la construcción de una matriz de distancias euclídeas.

2. Los elementos de la matriz de distancias euclídeas son equivalentes, o bastante aproximados, a los elementos δ_{rs} de **D**.

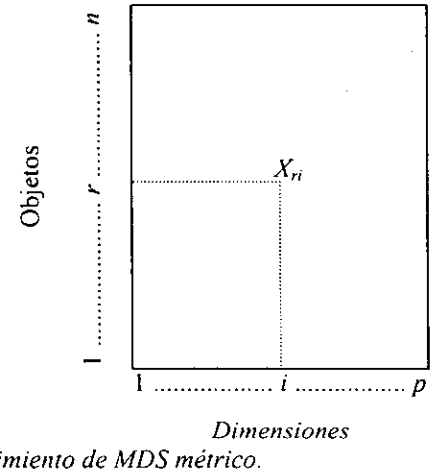


Figura 6.30. Salida del procedimiento de MDS métrico.

Los pasos a seguir para llevar a cabo un MDS métrico son los siguientes (Jobson, 1992) y (Cox y Cox, 1994):

PASO 1: Conversión de similitudes a disimilitudes

Las medidas de pares empleadas en la metodología de validación representan similitudes. Como el MDS métrico se basa en medidas de disimilitud es necesario realizar una transformación de la matriz **C** de similitudes en una matriz **D** de disimilitudes. Esta transformación se puede realizar mediante la siguiente ecuación:

$$\delta_{rs} = 1 - c_{rs}$$

equ. 6.52

Si se cumple que $c_{rs} \leq 1$ para todo r y s, que $c_{rr} = 1$ y que $c_{rs} = c_{sr}$ entonces se cumple que $\delta_{rs} \geq 0$, $\delta_{rr} = 0$ y $\delta_{rs} = \delta_{sr}$ lo que significa que la matriz **D** es una matriz de distancias (según las habíamos definido en el apartado 6.2.2.4).

PASO 2: Construcción de una matriz semidefinida positiva basada en *D*

El siguiente paso en el MDS métrico es convertir la matriz **D** en una matriz semidefinida positiva. Decimos que una matriz **A**, simétrica de orden ($n \times n$), es semidefinida positiva cuando se cumple que su *forma cuadrática*, definida como el escalar

$$\mathbf{x'Ax} = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} x_i x_j$$

equ. 6.53

es mayor o igual que cero para todo **x** e igual a cero para al menos un **x**. Siendo **x** un vector de orden ($n \times 1$) y **x'** su vector traspuesto.

Una matriz de disimilitudes **D** con ceros en la diagonal principal no es positiva semidefinida. Sin embargo se puede construir una matriz **A** ($n \times n$) que si lo sea basada en los elementos δ_{rs} de **D**. Los elementos a_{rs} de la nueva matriz **A** pueden ser determinados utilizando la siguiente relación:

$$a_{rs} = -\frac{1}{2}\left[\delta_{rs}^2 - \delta_{r.}^2 - \delta_{.s}^2 + \delta_{..}^2\right] \quad r, s = 1, 2, ..., n$$

equ. 6.54

donde

$$\delta_{r.}^2 = \frac{1}{n}\sum_{s=1}^{n} \delta_{rs}^2$$

$$\delta_{.s}^2 = \frac{1}{n}\sum_{r=1}^{n} \delta_{rs}^2$$

$$\delta_{..}^2 = \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} \delta_{rs}^2$$

La matriz **A** nos permite hallar la solución al MDS basándonos en los que se conoce como el *teorema fundamental del MDS*. Este teorema establece que si la matriz de distancias **D** es euclídea (es decir, es representable en un espacio euclídeo) entonces la matriz **A** construida a partir de la transformación definida en la equ. 6.54 es semidefinida positiva.

PASO 3: Obtención de las coordenadas a partir de la matriz A

La matriz **A** puede expresarse como **A** = **XX'** en donde **X** es la matriz ($n \times p$) de coordenadas. El rango de **A**, $r(\mathbf{A})$, es entonces

$$r(\mathbf{A}) = r(\mathbf{XX'}) = r(\mathbf{X}) = p$$

Como **A** es simétrica, positiva semidefinida (si **D** era euclídea) y de rango p, entonces tiene p autovalores no negativos y $n - p$ autovalores iguales a cero. Esto implica que la matriz **A** puede expresarse como $\mathbf{A} = \mathbf{V}\,\Lambda\,\mathbf{V}'$ en donde $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n]$ es la matriz de autovectores de **A** y $\Lambda = \text{diag}\,(\lambda_1, \lambda_2, ..., \lambda_n)$ es la matriz diagonal de autovalores. Se supone que $\mathbf{v}_i'\mathbf{v}_i = 1$ y, por conveniencia, que $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n \geq 0$.

Como existen $n - p$ autovalores iguales a cero **A** puede ser reescrita como $\mathbf{A}=\mathbf{V}_1\Lambda_1\mathbf{V}_1'$ en donde $\mathbf{V}_1 = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_p]$ y $\Lambda_1 = \text{diag}\,(\lambda_1, \lambda_2, ..., \lambda_p)$. De esta forma podemos hallar la matriz de coordenadas **X** como

$$\mathbf{X} = \mathbf{V}_1\Lambda_1^{1/2}$$

equ. 6.55

El vector de coordenadas del elemento i será

$$\mathbf{x}_i = \left(\mathbf{v}_{i1}\sqrt{\lambda_1}, \mathbf{v}_{i2}\sqrt{\lambda_2}, ..., \mathbf{v}_{ip}\sqrt{\lambda_p}\right)$$

Cada una de las variables tendrá media cero y serán únicas a no ser que sean multiplicadas por una determinada constante.

Matrices de distancias no euclídeas

En apartados anteriores habíamos visto como, si la matriz **D** de disimilitudes era euclídea, la matriz **A**, obtenida de **D** a partir de la equ. 6.54, era semidefinida positiva con rango p, con p autovalores positivos y $n - p$ autovalores nulos.

Sin embargo, ¿que pasa si la matriz **D** no representa una matriz de distancias euclídeas?, por ejemplo, debido a errores o subjetividades a la hora de establecer las distancias. En ese caso, **A** no sería semidefinida positiva y algunos de sus autovalores serían negativos, lo que nos llevaría a incluir valores complejos en las coordenadas de los elementos. Llegados a este punto hay dos posibles soluciones:

1) Añadir una determinada constante c a los elementos de **D** que no forman parte de la diagonal principal para convertir a **D** en una matriz euclídea.

2) Ignorar los autovalores negativos y seguir el procedimiento con el resto de autovalores.

El primer punto es lo que se conoce como el *problema de la constante aditiva* (additive constant problem) y consiste en que se debe definir una constante c lo suficientemente grande para asegurar que **D** sea euclídea, pero lo suficientemente pequeña para que no se incrementen excesivamente el número de dimensiones necesarias para representar a los datos. Algunas rutinas de MDS realizan aproximaciones al valor de c y Cailliez (1983) elaboró un método para determinarlo.

El segundo punto plantea una solución más sencilla basándose en el hecho de que, aunque exista una representación euclídea de los elementos en p dimensiones, si el número p es demasiado alto (valores mayores que tres) generalmente lo que se hace es representar únicamente la dos o tres dimensiones mayores. De esta forma, la claridad de la representación aumenta aunque se desprecie algo de exactitud. Por eso, si la matriz **A** presenta autovalores negativos, estos pueden ser descartados de forma que nos

quedemos con los dos o tres autovalores mayores (que nos conducirán a una representación en dos o tres dimensiones). Si los valores de los autovectores descartados no son demasiado elevados la representación obtenida será adecuada.

Validación de una solución de MDS métrico

Como hemos visto en el apartado anterior, al forzar que la representación MDS se realice en dos o tres dimensiones, probablemente estemos añadiendo un factor de error. La mejor forma de medir este error en un MDS métrico consiste en realizar una correlación lineal (ver equ. 6.36) entre las distancias iniciales δ_{ij} representadas en **D** y las distancias euclídeas, d_{ij}, obtenidas a partir de las coordenadas resultantes del análisis MDS de la siguiente forma

$$d_{ij} = \sqrt{\sum_{m=1}^{p} (x_{im} - x_{jm})^2}$$

equ. 6.56

siendo p el número de dimensiones. Usando el álgebra de matrices la equ. 6.56 puede reescribirse como

$$d_{ij} = \left[(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)'\right]^{1/2}$$

equ. 6.57

en donde $\mathbf{x}_i$ es el vector que contiene las p coordenadas del elemento i-ésimo.

Para el MDS no métrico es común usar la medida de STRESS vista en la equ. 6.51 como índice que nos indique la validez del análisis MDS.

Ejemplo de MDS métrico

Para ilustrar el procedimiento de MDS descrito consideraremos la matriz de similitudes (**C**) para el porcentaje de acuerdo utilizada en el análisis cluster (Tabla 7.16) y que reproducimos a continuación:

C	A	B	C	D	SE
A	1	0.8	0.6	0.4	0.7
B	0.8	1	0.4	0.2	0.7
C	0.6	0.4	1	0.6	0.4
D	0.4	0.2	0.6	1	0.3
SE	0.7	0.7	0.4	0.3	1

La matriz **C** representa similitudes. Para poder aplicar el MDS métrico es necesario convertirla en una matriz **D** de disimilitudes aplicando la equ. 6.52. El resultado es el siguiente:

D	A	B	C	D	SE
A	0	0.2	0.4	0.6	0.3
B	0.2	0	0.6	0.8	0.3
C	0.4	0.6	0	0.4	0.6
D	0.6	0.8	0.4	0	0.7
SE	0.3	0.3	0.6	0.7	0

Tabla 6.29. Matriz de disimilitudes obtenida de la matriz de similitudes de la Tabla 7.16.

Una vez construida la matriz de disimilitudes es necesario construir la matriz **A** que nos permita hallar las coordenadas de los distintos elementos. Los elementos de la matriz **A** se obtienen a partir de los elementos de **D** según se especifica en la equ. 6.54. El resultado obtenido es el siguiente:

A	A	B	C	D	SE
A	0.020	0.048	-0.021	-0.060	0.013
B	0.048	0.116	-0.073	-0.152	0.061
C	-0.021	-0.073	0.098	0.079	-0.083
D	-0.060	-0.152	0.079	0.220	-0.087
SE	0.013	0.061	-0.083	-0.087	0.096

Tabla 6.30. Matriz semidefinida positiva obtenida a partir de la matriz de disimilitudes de la Tabla 6.29.

Si la matriz **A** fuera semidefinida positiva de rango p, presentaría p autovalores positivos y el resto a cero. En caso contrario aparecerán autovalores negativos. Aplicando el algoritmo de Jacobi a la matriz **A** podemos hallar sus autovectores y autovalores obteniéndose los valores de la Tabla 6.31.

V	v_1	v_2	v_3	v_4	v_5
A	0.793	0.447	0.215	-0.184	-0.301
B	-0.581	0.447	0.210	-0.499	-0.413
C	-0.129	0.447	0.605	0.364	0.534
D	-0.122	0.447	-0.430	0.676	-0.379
SE	0.039	0.447	-0.600	-0.357	0.558
Autovalores	-0.002	0	0.091	0.437	0.023

Tabla 6.31. Autovalores y sus correspondientes autovectores para los datos de la Tabla 6.30.

Como vemos, la matriz **A** no es semidefinida positiva porque presenta un autovalor negativo. Esto implica que la matriz de disimilitudes **D** no es directamente representable en un espacio euclídeo. Sin embargo, podemos escoger las coordenadas representadas por los dos mayores autovalores y sus correspondientes autovectores (marcadas en gris) para realizar nuestra representación. El error cometido no será demasiado elevado porque los valores de los autovalores rechazados son poco significativos. Las coordenadas resultantes de aplicar la equ. 6.55 se muestran en la siguiente tabla (siendo $\mathbf{x}_1$ las coordenadas que corresponden al mayor autovalor y $\mathbf{x}_2$ las que corresponden al segundo mayor autovalor).

X	x_1	x_2
A	-0.122	0.065
B	-0.330	0.063
C	0.241	0.183
D	0.447	-0.130
SE	-0.236	-0.181

Tabla 6.32. Coordenadas bidimensionales resultantes de aplicar la equ. 6.55 a los autovalores y autovectores de la Tabla 6.31.

La representación de los distintos expertos en un espacio bidimensional según estas coordenadas puede verse en la Figura 6.31. Es importante destacar que la ordenación en los ejes es arbitraria, pudiendo rotarse la configuración de la forma que más convenga. También puede verse que la media de las coordenadas $\mathbf{x}_1$ y $\mathbf{x}_2$ es cero.

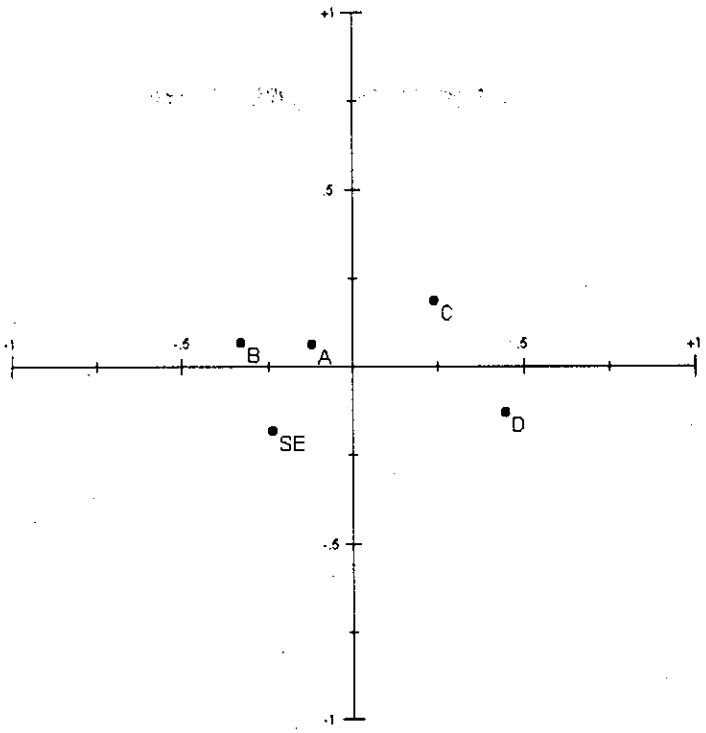


Figura 6.31. Solución del MDS métrico para los datos del porcentaje de acuerdo.

El último paso que quedaría sería validar si la solución mostrada es adecuada, recordemos que hemos eliminado algunas coordenadas para poder representar a los expertos en un espacio bidimensional. Para ello en primer lugar hallaremos las distancias euclídeas existentes entre los expertos a partir de las coordenadas de la matriz **X**. Estas distancias son:

	A	B	C	D	SE
A	0	0.208	0.382	0.601	0.271
B	0.208	0	0.583	0.801	0.261
C	0.382	0.583	0	0.375	0.600
D	0.601	0.801	0.375	0	0.685
SE	0.271	0.261	0.600	0.685	0

Tabla 6.33. Distancias euclídeas derivadas de las coordenadas de la Tabla 6.32.

A partir de estas distancias podemos realizar una correlación lineal con las disimilitudes originales que se utilizaron para realizar el MDS métrico. El resultado debe ser cercano a 1 para considerar que la representación bidimensional es adecuada. En este caso, el valor del índice de correlación es 0.998 lo que implica que las diferencias existentes entre las distancias euclídeas y las disimilitudes iniciales no son demasiado significativas.

6.2.4. Medidas de dispersión y tendencia

Las medidas de dispersión y tendencia son medidas de grupo que se encargan de analizar respectivamente, la dispersión de los resultados de un determinado experto en comparación con los resultados del resto de expertos, y hacia qué categorías tienden a desviarse dichos resultados.

Las medidas de dispersión y tendencia, originalmente por Lucien Duckstein, fueron utilizadas por Bahill et al. (1995) para analizar los resultados de un sistema de apoyo a la decisión y cuatro expertos humanos en la detección precoz del tartamudeo.

6.2.4.1. Medida de dispersión

La medida de dispersión para un experto k se define como

$$\text{Dispersion}_k = \frac{1}{n_c}\sum_{j=1}^{n_c}\left(\sqrt{\frac{1}{n_e - 1}\sum_{i=1}^{n_e}\left(D_{kj} - D_{ij}\right)^2}\right)$$

equ. 6.58

en donde n_c representa el número de casos considerados, n_e el número de expertos y D_{ij} el número de orden del diagnóstico realizado por el experto i sobre el caso j (las categorías del diagnóstico siguen una escala ordinal a las que se ha asignado un determinado número de orden, por ejemplo, 1 = "Muy Bajo", 2 = "Muy Alto", etc.).

Como vemos en la equ. 6.58 la medida de dispersión para un experto k es la media de la dispersión existente en los distintos casos de la base de datos. La dispersión en un caso específico se halla como la media de las diferencias cuadráticas entre los diagnósticos del experto k y el resto de expertos (después se realiza una raíz cuadrada para que la medida resultante tenga las mismas dimensiones que los datos originales).

Volvamos a considerar la base de datos de validación de la Tabla 6.2.

Casos	A	B	C	D	SE
1	ALTO	ALTO	ALTO	ALTO	ALTO
2	ALTO	BAJO	ALTO	ALTO	BAJO
3	BAJO	BAJO	NORMAL	NORMAL	BAJO
4	NORMAL	BAJO	NORMAL	NORMAL	NORMAL
5	MUY ALTO	MUY ALTO	ALTO	ALTO	MUY ALTO
6	BAJO	BAJO	BAJO	BAJO	NORMAL
7	MUY BAJO	MUY BAJO	NORMAL	BAJO	BAJO
8	NORMAL	NORMAL	NORMAL	ALTO	NORMAL
9	NORMAL	NORMAL	BAJO	MUY BAJO	NORMAL
10	BAJO	BAJO	BAJO	ALTO	BAJO

Si queremos hallar la dispersión de los resultados del experto SE es necesario hallar la dispersión de cada uno de los 10 casos. Por ejemplo, para el caso 1 la dispersión es cero, para el caso dos, suponiendo que 1 = "Muy Bajo", 2 = "Bajo", 3 = "Normal", 4 = "Alto" y 5 = "Muy Alto", la dispersión es:

$$\sqrt{\frac{(2-4)^2 + (2-2)^2 + (2-4)^2 + (2-4)^2}{4}} = \sqrt{\frac{4+0+4+4}{4}} = \sqrt{\frac{12}{4}} = \sqrt{3} = 1.732$$

Siguiendo con el resto de casos obtenemos la Tabla 6.34 que indica la dispersión de los resultados del sistema experto para cada caso.

Casos	$Dispersión_{SE}$
1	0
2	1.732
3	0.707
4	0.5
5	0.707
6	1
7	0.866
8	0.5
9	1.118
10	1
Total	8.130
Media	0.813

Tabla 6.34. Dispersión de los datos del sistema experto para cada caso de la base de datos de validación.

La media de estos valores, 0.813, nos indicará la medida de dispersión de los datos del sistema experto para nuestra base de datos. Es importante tener en cuenta que todos los expertos son considerados de la misma categoría.

La medida de dispersión para todos los expertos involucrados en nuestro estudio sería

Casos	Dispersión
A	0.767
B	0.849
C	0.823
D	0.982
SE	0.813

Tabla 6.35. Medida de dispersión para los expertos humanos y el sistema experto.

Como vemos los datos concuerdan bastante bien con la configuración que habíamos obtenido del MDS (Figura 6.31) en la que el experto A se situaba en el centro del resto de expertos (y por lo tanto es el que presenta menor dispersión) y el experto más alejado de todos era el experto D (al cual corresponde la máxima dispersión).

6.2.4.2. Medida de tendencia

La medida de tendencia intenta mostrar si la magnitud de los resultados de un experto en particular tiende a ser menor o mayor que la magnitud de los resultados del resto de expertos. La medida de tendencia para un experto k se define de forma muy similar a la medida de dispersión

$$\text{Tendencia}_k = \frac{1}{n_c}\sum_{j=1}^{n_c}\left(\frac{1}{n_e-1}\sum_{i=1}^{n_e}\left(D_{kj}-D_{ij}\right)\right) \qquad \text{equ. 6.59}$$

en donde n_c, n_e y D_{ij} representan lo mismo que en la medida de dispersión.

En este caso vemos que también se realiza la media de las tendencias mostradas para cada caso, definiéndose la tendencia de cada caso como la media de las diferencias existentes entre el diagnóstico del experto tomado en consideración y los diagnósticos del resto de expertos (al no elevar el valor de las diferencias al cuadrado podemos encontrarnos con valores negativos).

Por ejemplo para el segundo caso la medida de tendencia para el sistema experto sería:

$$\frac{(2-4)+(2-2)+(2-4)+(2-4)}{4} = \frac{(-2)+(0)+(-2)+(-2)}{4} = \frac{-6}{4} = -1.5$$

Para el resto de casos la medida de tendencia sería:

Casos	$Tendencia_{SE}$
1	0
2	-1.5
3	-0.5
4	0.25
5	0.5
6	1
7	0.25
8	-0.25
9	0.75
10	-0.5
Total	0
Media	0

Tabla 6.36. Dispersión de los datos del sistema experto para cada caso de la base de datos de validación.

La media de estos valores, 0, nos indicará la medida de tendencia de los datos del sistema experto para nuestra base de datos. Para el resto de expertos involucrados en nuestro estudio su valor sería

Casos	Tendencia
A	0
B	-0.375
C	0.125
D	0.250
SE	0

Tabla 6.37. Medida de dispersión para los expertos humanos y el sistema experto.

Esta medida es más fácil de interpretar si atendemos a los datos de la Figura 6.32 en la que representamos los diagnósticos de los expertos en una gráfica XY en la que las Xs representan los casos y las Ys los posibles diagnósticos.

La gráfica, a pesar de ser un poco confusa, muestra como el experto que presenta valores más bajos en sus interpretaciones para la mayoría de los casos es el experto B (la medida de tendencia tomará valores negativos). Por otro lado, el experto D presenta valores más altos en sus interpretaciones que el resto de expertos (la medida de tendencia es positiva). De todas formas las diferencias no son demasiado elevadas y los valores de la medida de tendencia son pequeños.

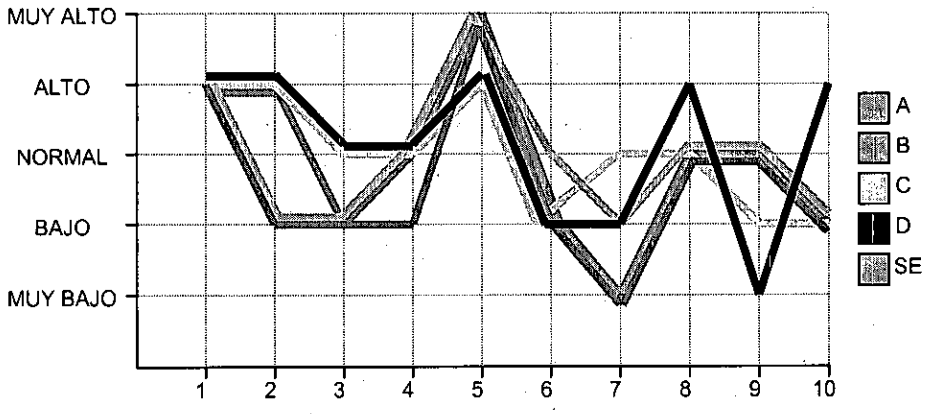


Figura 6.32. Representación de los datos de los expertos como líneas en una gráfica XY

6.3. Ratios de acuerdo

El tabaco es una de las principales causas de las estadísticas.
Anónimo

El Instituto Schwine-Kitzenger estudio 47 hombres de más de 100 años y descubrió que tenían las siguientes características en común: (1) Todos tenían apetitos moderados, (2) todos provenían de la clase media y (3) todos menos dos estaban muertos.
Anónimo

La muerte de un soldado ruso es una tragedia, la muerte de un millón de ellos es una estadistica
Joseph Stalin (Político soviético. 1879 – 1953)

El último tipo de medidas cuantitativas incluidas dentro de nuestra metodología de validación lo constituyen los llamados *ratios de acuerdo*. Los ratios de acuerdo se encargan de medir el acuerdo existente entre un experto (o sistema experto) y una referencia estándar. Dicha referencia puede ser un consenso existente entre los expertos, lo que implicaría una validación contra el experto, o la solución real al problema planteado, lo que implicaría una validación contra el problema.

Evidentemente, el sistema experto puede compararse con una referencia estándar a través de los tests de pares, descritos en el capítulo 6.1, para hallar sus niveles de acuerdo o asociación. Quizá la única diferencia existente con los tests de pares realizados entre dos expertos es la disposición de los pesos de desacuerdo que veíamos en la medida kappa ponderada. Así, al ser los dos expertos de la misma cualificación es normal disponer los pesos de forma simétrica. Sin embargo, si uno de los expertos se substituye por una referencia estándar, esta simetría no está tan clara, ya que ahora sabemos cual es la respuesta correcta, y podemos determinar la importancia del error cometido.

La diferencia fundamental existente entre los tests de pares y los ratios de acuerdo es que los primeros tratan el diagnóstico en su conjunto, mientras que los segundos analizan los resultados obtenidos en las distintas categorías en las que se divide el diagnóstico.

6.3.1. Cálculo de los ratios de acuerdo

El cálculo de los ratios de acuerdo se basa en la construcción de una tabla de contingencia 2 × 2 para cada una de las categorías en las que se divide el diagnóstico (Tabla 6.38). En esta tabla se relacionan los resultados del experto (o sistema experto) con los resultados de la referencia estándar para esa categoría en particular (Adlassnig y Scheithauer, 1989).

		Referencia Estándar	
		D	$\neg D$
Sistema Experto	D	a	b
	$\neg D$	c	d
		$a + c$	$b + d$

Tabla 6.38. Tabla de contingencia 2 × 2 para el cálculo de los ratios de acuerdo. D representa la presencia de una decisión o categoría diagnóstica, mientras que ¬D representa su ausencia.

Los ratios de acuerdo que se pueden definir sobre esta tabla son los siguientes:

- *Tasa de verdaderos positivos*: Se conoce también como *sensibilidad* y se calcula como $a / (a + c)$. Representa el número de veces que hemos diagnosticado correctamente la categoría en consideración dividido por el número total de veces que aparecía dicha categoría en el estándar.

- *Tasa de verdaderos negativos*: Se conoce también como *especificidad* y se calcula como $d / (b + d)$. Representa el número de veces que hemos diagnosticado correctamente la ausencia de la categoría seleccionada dividido por el número de veces que dicha categoría no aparecía en el estándar.

- *Tasa de falsos positivos*: Se representa por $b / (b + d)$ o también como (1 – especificidad). Representa el número de veces que erróneamente hemos diagnosticado la categoría en consideración partido por el número de veces que dicha categoría no aparecía en el estándar.

- *Tasa de falsos negativos*: Se representa por $c / (a + c)$ o también como (1 – sensibilidad). Representa el número de veces que erróneamente no hemos diagnosticado la categoría en consideración partido por el número de veces que dicha categoría aparecía en el estándar.

En la base de datos de validación de la Tabla 6.2 se incluían cuatro expertos humanos y un sistema experto. Si suponemos, por ejemplo, que el experto que tiene una mayor categoría profesional y experiencia es el experto A podemos tomar sus resultados como una referencia estándar. En base a ello podemos analizar los resultados del sistema experto con los resultados de dicho experto.

Los resultados de los tests de pares para este par de expertos son:

MEDIDAS DE ACUERDO		MEDIDAS DE ASOCIACIÓN	
Kappa	0.589	Tau de Kendall	0.489
Kappa ponderada	0.725	Tau b de Kendall	0.621
Porcentaje de acuerdo	0.700	Gamma de Goodman-Kruskal	0.733
Porcentaje de acuerdo dentro de uno	0.900	Rho de Spearman	0.655

Tabla 6.39. Resultados de los tests de pares para el experto A y el sistema experto.

Para hallar los ratios de acuerdo es necesario construir 5 tablas de contingencia (una para cada uno de los posibles diagnósticos) y luego calcular los ratios de la forma que hemos descrito.

		Referencia Estándar	
		MUY BAJO	¬ MUY BAJO
Sistema Experto	MUY BAJO	0	0
	¬ MUY BAJO	1	9
		1	9

TP = 0
TN = 1
FP = 0
FN = 1

Tabla 6.40. Ratios de acuerdo para la categoría MUY BAJO.

		Referencia Estándar	
		BAJO	¬ BAJO
Sistema Experto	BAJO	2	2
	¬ BAJO	1	5
		3	7

TP = 0.667
TN = 0.714
FP = 0.286
FN = 0.333

Tabla 6.41. Ratios de acuerdo para la categoría BAJO.

		Referencia Estándar	
		NORMAL	¬ NORMAL
Sistema Experto	NORMAL	3	1
	¬ NORMAL	0	6
		3	7

TP = 1
TN = 0.857
FP = 0.143
FN = 0

Tabla 6.42. Ratios de acuerdo para la categoría NORMAL.

		Referencia Estándar	
		ALTO	¬ ALTO
Sistema Experto	ALTO	1	0
	¬ ALTO	1	8
		2	8

TP = 0.5
TN = 1
FP = 0
FN = 0.5

Tabla 6.43. Ratios de acuerdo para la categoría ALTO.

		Referencia Estándar	
		MUY ALTO	¬ MUY ALTO
Sistema Experto	MUY ALTO	1	0
	¬ MUY ALTO	0	9
		1	9

TP = 1
TN = 1
FP = 0
FN = 0

Tabla 6.44. Ratios de acuerdo para la categoría MUY ALTO.

Un problema común con los ratios de acuerdo consiste en que, si normalmente la casuística de validación es escasa, esta complicación se agrava al evaluar las categorías por separado (ya que puede haber categorías en las cuales no haya suficientes casos para desarrollar un validación fiable). Así, por ejemplo, el sistema experto nunca diagnóstica la categoría MUY BAJO, y el experto A sólo lo hace en un único caso.

Generalmente lo que se busca es que la tasa de verdaderos positivos y verdaderos negativos sea lo más alta posible, mientras que la tasa de falsos positivos y falsos negativos sea lo más baja posible.

6.3.2. Medidas de similitud

Además de los ratios de acuerdo se incluyen dos medidas de similitud, el porcentaje de acuerdo y la medida de Jaccard, que son los coeficientes más usados en la práctica en tablas de contingencias 2 × 2 (Everitt, 1993).

El porcentaje de acuerdo se calcula en la tabla 2 × 2 como

$$\text{Porcentaje} = \frac{a+d}{a+b+c+d}$$

equ. 6.60

y representa el porcentaje de casos en los que el experto ha coincidido con el estándar para la categoría tomada en consideración. Es importante no confundirlo con el porcentaje de acuerdo de los tests de pares, en los que se consideraban todas las posibles categorías.

Por otro lado también se utiliza la medida de Jaccard que se define como

$$\text{Jaccard} = \frac{a}{a+b+c}$$

equ. 6.61

La medida de Jaccard elimina, tanto del numerador como del denominador, el número de casos en los que el experto y el estándar no han diagnosticado la categoría considerada (es decir, eliminan el número *d*).

Los resultados de aplicar estas medidas de similitud al ejemplo visto en los ratios de acuerdo se muestra en las siguientes tablas

		Referencia Estándar	
		MUY BAJO	¬ MUY BAJO
Sistema Experto	MUY BAJO	0	0
	¬ MUY BAJO	1	9
		1	9

Porcentaje = 0.9
Jaccard = 0

Tabla 6.45. Porcentaje de acuerdo y medida de Jaccard para la categoría MUY BAJO.

		Referencia Estándar	
		BAJO	¬ BAJO
Sistema Experto	BAJO	2	2
	¬ BAJO	1	5
		3	7

Porcentaje = 0.7
Jaccard = 0.4

Tabla 6.46. Porcentaje de acuerdo y medida de Jaccard para la categoría BAJO.

		Referencia Estándar	
		NORMAL	¬ NORMAL
Sistema Experto	NORMAL	3	1
	¬ NORMAL	0	6
		3	7

Porcentaje = 0.9
Jaccard = 0.75

Tabla 6.47. Porcentaje de acuerdo y medida de Jaccard para la categoría NORMAL.

		Referencia Estándar	
		ALTO	¬ ALTO
Sistema Experto	ALTO	1	0
	¬ ALTO	1	8
		2	8

Porcentaje = 0.9
Jaccard = 0.5

Tabla 6.48. Porcentaje de acuerdo y medida de Jaccard para la categoría ALTO.

		Referencia Estándar	
		MUY ALTO	¬ MUY ALTO
Sistema Experto	MUY ALTO	1	0
	¬ MUY ALTO	0	9
		1	9

Porcentaje = 1
Jaccard = 1

Tabla 6.49. Porcentaje de acuerdo y medida de Jaccard para la categoría MUY ALTO.

La medida de Jaccard es muy útil para medir acuerdos en aquellas situaciones en las que resultan más importantes los resultados positivos que los negativos.

Otras medidas de similitud en tablas 2 × 2 pueden encontrarse en (Cox y Cox, 1994) y (Enc. Stat. Sci., 1981).

6.4. Resumen

En este capítulo hemos detallado las principales características de una serie de medidas estadísticas potencialmente útiles en el proceso de validación. Estas medidas se dividen en tres grupos principales: tests de pares, tests de grupo y ratios de acuerdo.

Los tests de pares se utilizan para medir el acuerdo o la asociación entre un par de expertos. Dentro de las medidas de acuerdo incluimos el porcentaje de acuerdo, el porcentaje de acuerdo dentro de uno, el índice kappa y el índice kappa ponderada. Estos dos últimos han sido utilizados a menudo en la validación de sistemas expertos porque permiten eliminar aquellos acuerdos que son debidos a la casualidad. Dentro de los tests de pares también se incluyen medidas de asociación como la tau y la tau b de Kendall, la gamma de Goodman-Kruskal o la rho de Spearman.

Los tests de pares se utilizan como base para el desarrollo de las medidas de grupo, estas últimas pretenden organizar al grupo de expertos según la similitud de sus interpretaciones. Dentro de las medidas de grupo podemos destacar las medidas de Williams, el análisis cluster, el escalamiento multidimensional y las medidas de dispersión y tendencia (que son las únicas que no se basan en los resultados de los tests de pares).

Por último incluimos los ratios de acuerdo, que son utilizados para comparar el rendimiento de un determinado experto frente a una referencia estándar.

Las características particulares de todas estas medidas, cuando son utilizadas en entornos de validación, se estudiará en el próximo capítulo en el que expondremos nuestra propuesta a una metodología de validación.

7. METODOLOGÍA PROPUESTA PARA LA VALIDACIÓN DE SISTEMAS INTELIGENTES

Podemos alcanzar la sabiduría por tres métodos: Primero, la reflexión, que es el más noble; segundo, la imitación, que es el más sencillo; y tercero, la experiencia, que es el más amargo.
Confucio (Filósofo chino. 551-479 a.C.)

El arte y la ciencia tienen su punto de encuentro en los métodos.
Edward Bulwer-Lytton.

PLAN: Preocuparse por el mejor método para conseguir un resultado accidental.
"The Devil's Dictionary", 1911
Ambrose Bierce (Escritor y periodista estadounidense. 1842-1914).

Uno de los principales problemas que surge a la hora de realizar la validación de un sistema experto es la falta de una metodología estándar. Generalmente todas las validaciones se suelen realizar de una manera informal que dificulta su interpretación y comparación.

Este trabajo propone una metodología que se basa en la caracterización del proceso de validación, tal y como hemos visto en el capítulo 5, y en la utilización de medidas cuantitativas, aunque no descarta la realización de medidas cualitativas.

La metodología se compone de tres partes claramente diferenciadas: planificación, aplicación e interpretación (Figura 7.1), que desarrollaremos a continuación.

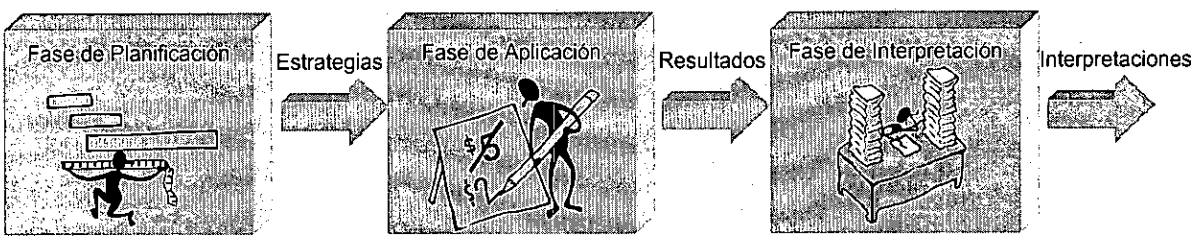


Figura 7.1. Fases en la metodología de validación de sistemas expertos.

7.1. Fase de planificación

La primera fase de nuestra metodología de validación es la fase de planificación. Esta fase es necesaria porque, el dominio de aplicación del sistema, las propias características del sistema y la fase de desarrollo en la que se encuentra, pueden motivar la utilización de determinados paradigmas de validación en detrimento de otros.

La fase de planificación es una fase netamente heurística en la que, la experiencia previa del ingeniero de conocimiento, es determinante para el correcto establecimiento de un plan de validación. El esquema de esta fase se puede ver en la Figura 7.2.

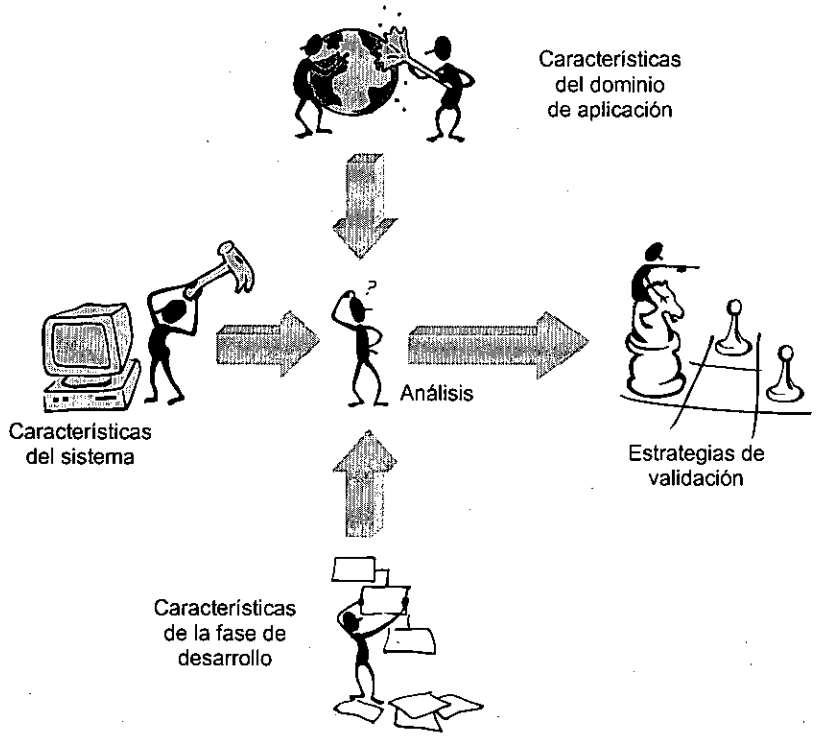


Figura 7.2. Estructura de la fase de planificación en la validación de sistemas expertos.

7.1.1. Influencia del dominio de aplicación

Las características del dominio de aplicación en el que se va a utilizar el sistema tienen un importancia fundamental a la hora de determinar los procesos de validación.

Dominios críticos

El principal inconveniente con que se enfrentan la mayoría de los sistemas expertos y que, en cierto sentido, los aleja del software convencional, es que en la mayoría de las ocasiones se utilizan en entornos denominados críticos. En dichos dominios el coste de una decisión errónea es muy elevado por lo que el proceso de validación debe ser más riguroso.

Además, un dominio crítico puede limitar las técnicas a emplear en la validación. De esta forma, la realización de pruebas prospectivas y tests de campo están muy limitadas cuando nos movemos en dichos entornos, y sólo podremos realizarlas en sistemas que no exijan una manipulación del entorno (por ejemplo sistemas de predicción).

Criterio de validación

Existen dos posibles criterios para validar un sistema experto, una validación contra los expertos y una validación contra el problema. Sin embargo, la mayoría de las veces el propio dominio de aplicación limita la posibilidad de elección del criterio de validación.

En dominios en la que la disponibilidad de los expertos es escasa, es difícil conseguir a un grupo de ellos para que colaboren en la validación de nuestro sistemas. En tales casos lo más probable será contar con un único experto, por lo que la objetividad de nuestro estudio puede quedar en entredicho. Si la disponibilidad de

expertos no es un problema, lo más adecuado es realizar la validación contra un grupo de expertos, o utilizar dicho grupo de expertos para realizar un consenso que pueda ser utilizado como estándar en la validación.

En cuanto a la validación contra el problema, no siempre es posible obtener un estándar que nos indique que el problema se ha resuelto correctamente. La validación orientada al problema se utiliza sobre todo en tareas de pronóstico, en donde es posible conocer si el resultado del pronóstico fue correcto o no.

Perfil del usuario final

En el proceso de validación pueden estar involucrados: el ingeniero del conocimiento, expertos del dominio, evaluadores independientes y usuarios finales. Los usuarios finales no suelen tener una participación activa en el proceso de validación orientado a los resultados. Sin embargo, si los usuarios son expertos del domino puede ser común utilizarlos para realizar técnicas como los tests de campo. Si los usuarios no son expertos, normalmente sólo podrán colaborar en una validación orientada al uso.

Lamberti y Newsome (1989) descubrieron que el rendimiento de usuarios no expertos aumentaba cuando eran confrontados a cuestiones concretas en vez de a representaciones abstractas.

7.1.2. Influencia del sistema

Las características del sistema que va a ser validado también influyen en la forma a llevar a cabo dicha validación.

División en subsistemas

La validación de subsistemas sólo podrá llevarse a cabo si es posible dividir nuestro sistema experto en módulos independientes. Si los módulos actúan como salidas y entradas unos de otros, o si su interacción es elevada, puede no ser posible validarlos por separado.

Manejo de incertidumbre

Si el sistema maneja medidas de certidumbre es necesario contemplarlas en el proceso de validación. Una forma típica de realizar la validación en presencia de medidas de certidumbre es efectuando estudios de sensibilidad, en los que se realizan pequeños cambios en los datos de entrada y se estudia su efecto en las salidas del sistema experto.

Tipo de las variables de salida

Las interpretaciones obtenidas por el sistema experto también influyen en la metodología de validación. Si atendemos a la bibliografía vemos que existen dos posibles tipos para las variables de salida (Figura 7.3): categóricas (que se subdividen en nominales y ordinales) y numéricas (que se subdividen por un lado en escalas de intervalos y escalas de ratios, y por otro lado en datos discretos o continuos).

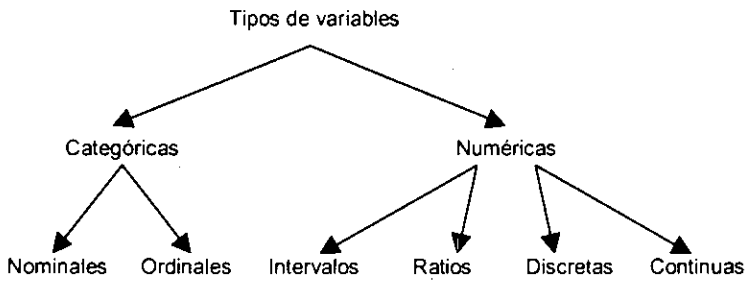


Figura 7.3. Clasificación de variables

Las *variables categóricas* son aquellas que asignan, a cada miembro de la población, una determinada categoría semántica de un conjunto finito de posibles categorías. Este tipo de variables (también conocidas como variables cualitativas) se dividen a su vez en variables nominales y variables ordinales.

Decimos que una variable sigue una escala *nominal* cuando sus posibles valores se dividen en etiquetas semánticas, entre las que no puede establecerse ningún tipo de relación. Las escalas nominales se consideran las formas más "débiles" de medida, ya que son las que ofrecen menos información. Sin embargo, suelen ser comunes en los sistemas expertos. Por ejemplo, una escala nominal sería la identificación de un problema psiquiátrico, que puede recaer en una de las siguientes categorías: "Neurosis", "Psicosis" o "Desordenes de personalidad".

Una variable sigue una escala *ordinal* cuando puede establecerse una relación de orden entre los posibles valores que puede tomar, aunque la distancia entre dichos valores no sea conocida. Por ejemplo, una escala ordinal sería la identificación de un determinado parámetro como "Muy Alto", "Alto", "Normal", "Bajo" o "Muy Bajo". Las escalar ordinales proveen más información que las nominales.

Las *variables numéricas* (o cuantitativas) son aquellas que toman valores representados por un número significativo. Un ejemplo de este tipo de variables sería la edad. Las variables ordinales también pueden utilizar números para sus categorías (podemos hacer que "Muy Alto" sea 5, "Alto" sea 4, etc.), sin embargo la diferencia con las variables numéricas es que en las ordinales sólo conocemos el orden de las categorías pero no hay nada que indique su magnitud (podríamos utilizar cualquier otra combinación numérica siempre y cuando mantengamos el orden).

Las variable numéricas pueden dividirse a su vez en dos tipos de escalas: escalas de intervalos y escalas de ratios. Decimos que una variable numérica pertenece a una *escala de intervalos* cuando en dicha escala no existe un cero significativo. Un ejemplo de esto sería un sistema de predicción meteorológica, en la que una de sus salidas fuera la predicción de temperatura. La salida del sistema puede ser medida en grados Celsius (C) y en grados Fahrenheit (F). Ambas escalas tienen un cero pero dicho valor ha sido designado de forma arbitraria (la relación entre grados F y grados C es la siguiente $F = 32 + 1.8 \times C$ como se puede ver en la Tabla 7.1).

°C	°F
-30	-22
-15	5
0	32
15	59
30	86

Tabla 7.1. Relación entre los grados Celsius y Fahrenheit.

El hecho de que el cero haya sido fijado de forma arbitraria provoca que no tengan sentido hablar de que la temperatura de 30° C es el doble que la temperatura 15° C ya que si medimos en la escala Fahrenheit esto no se cumple (86° F no es el doble que 59° F). Lo que si tiene sentido es decir que la diferencia de temperatura, o el intervalo, que hay entre 0° C y 15° C es el mismo que hay entre 15° C y 30° C (la diferencia entre 32° F y 59° F es la misma que la existente entre 59° F y 86° F).

Por otro lado las *escalas de ratios* sí tienen un cero significativo. Así, supongamos que un sistema indica la administración de una determinada cantidad de un medicamento, esta cantidad puede ser expresada en gramos (gr) o en onzas (oz) pero el cero de ambas escalas es el mismo como se ve en la Tabla 7.2 (la relación entre gramos y onzas es la siguiente 1 oz = 28.35 gr).

gr	oz
0	0
10	0.35
20	0.70

Tabla 7.2. Relación entre los gramos y las onzas.

En este caso sí podemos decir que el peso de 20gr es el doble que el peso de 10gr (0.70oz también es el doble de 0.35oz).

Otra posible división de los datos numéricos es en datos discretos y datos continuos. Los *valores discretos* son múltiplos de una determinada cantidad indivisible y se suelen representar generalmente con números enteros (número de hijos de una familia, número de accidentes de tráfico en un año). Por otro lado, en *los valores continuos* no existe una unidad indivisible y las variables pueden tomar cualquier valor dentro del intervalo de referencia (como por ejemplo la altura, el peso, etc.)

Después de ver los distintos tipos de variables existentes veamos como afecta esta circunstancia a la validación de un sistema inteligente.

Si la salida del sistema es una variable nominal, no puede establecerse un orden entre las distintas categorías, y las discrepancias producidas entre dos categorías cualesquiera se tratarán de la misma forma. Generalmente el acuerdo se medirá a través del porcentaje de acuerdo o el índice kappa. También puede suceder que, aunque no podamos establecer un orden, existan discrepancias más graves que otras, en ese caso se recomienda la utilización del índice kappa ponderada. Medidas como el porcentaje de acuerdo dentro de uno o las medidas de asociación (coef. de correlación, tau, gamma y rho) no son aplicables a este tipo de variables porque presuponen un orden en las categorías.

Sin embargo, si son aplicables las llamadas medidas de *asociación predictiva*, entre las que incluimos la lambda (λ) de Guttman (1941) y Goodman-Kruskal (1954), la tau (τ) de Goodman-Kruskal (1954), y la eta (η) de Theil (1970). El objetivo de estas medidas consiste en comprobar en que grado la interpretación del sistema experto puede utilizarse para predecir la interpretación del experto. Entre los trabajos que utilizan este tipo de medidas encontramos la validación del sistema MEDAS (Georgakis et al., 1990), sin embargo, estas medidas son poco utilizadas y resulta más corriente uso de medidas de acuerdo en escalas nominales.

Si la salida del sistema es una variable ordinal, puede ser necesario tener en cuenta que cometer un error entre dos categorías muy distantes en la escala puede ser mucho más grave que cometerlo entre dos categorías adyacentes (no es lo mismo una discrepancia entre "Muy Alto" y "Muy Bajo", que una discrepancia entre "Muy Alto" y "Alto"). En tal caso se recomienda que se utilicen técnicas de validación que tengan en cuenta estas discrepancias (como el porcentaje de acuerdo dentro de uno o la medida kappa ponderada). Las medidas de asociación descritas dentro de los tests de pares son aplicables solamente a datos ordinales. De ellas, la tau de Kendall y sus medidas derivadas (tau b y gamma) son las que utilizan menos información, ya que se pueden obtener utilizando sólo la relación de orden entre las distintas categorías. El coeficiente de correlación y la rho de Spearman necesitan que las categorías sean representadas a través de números (que pueden ser valores numéricos o rangos).

Los valores nominales y ordinales son directamente representables en tablas de contingencia, a partir de las cuales podemos extraer las distintas medidas de acuerdo o asociación. Las salidas en formato numérico no pueden ser representadas a través de una tabla de contingencia, para que esto sea posible es necesario categorizarlas, dividiendo los posibles valores en intervalos y asignando etiquetas a dichos intervalos. En la Tabla 7.3 mostramos la categorización de los datos del pH para un paciente adulto normal.

Valores del pH	Etiqueta
pH < 7.30	Muy Bajo
7.30 ≤ pH ≤ 7.34	Bajo
7.34 < pH < 7.46	Normal
7.46 ≤ pH ≤ 7.50	Alto
7.50 < pH	Muy Alto

Tabla 7.3 Categorización semántica de los datos del pH para un paciente adulto normal.

Como vemos, mediante la categorización se convierten los datos numéricos en datos ordinales. Aunque esta técnica facilita el tratamiento de los datos, la pérdida de información que se produce al crear las categorías puede llevar al establecimiento de conclusiones erróneas. Por ello es necesario que el procedimiento de categorización semántica se haga de manera cuidadosa.

Pero no siempre es necesario categorizar los datos numéricos para poder tratarlos, pueden utilizarse técnicas para medir directamente las diferencias entre variables numéricas (como los coeficientes de exactitud propuestos por Shapiro (1977)), o distancias aritméticas para medir las diferencias existentes entre dos vectores numéricos (como pueden ser los vectores que incluyen las probabilidades de aparición de una serie de hipótesis distintas). Estas medidas ya fueron vistas en el capítulo 5.6.2 en el que tratábamos las medidas cuantitativas utilizadas en la validación.

Los datos numéricos proveen más información que los datos categóricos, pero no es frecuente que aparezcan en campos como la validación de las interpretaciones de sistemas expertos. De ellos, el que más información contiene son los datos medidos en una escala de ratio, ya que, como indica su nombre, permite las comparaciones en base a ratios (un valor es x veces mayor o menor que otro valor). Los datos de la escala intervalo solo permiten comparar diferencias de valores entre intervalos.

La división de los datos en discretos o continuos no tiene repercusiones en su interpretación, sino más bien en su tratamiento.

Tipo de problema tratado

El tipo de problema tratado también influye en la metodología de validación. Podemos distinguir dos tipos principales de problemas: (1) problemas de análisis, en los que a partir de una serie de casos los asignamos dentro de una determinada categoría (pueden ser problemas de diagnóstico, problemas de predicción, etc.) y (2) problemas de síntesis, en el que el resultado incluye la realización de un determinado plan de acción (por ejemplo, la elaboración de un determinado plan terapéutico).

Los sistemas que tratan problemas de análisis son más fáciles de validar ya que es posible aplicarlos sobre casos históricos y es más fácil de encontrar referencias estándar para dicha validación.

Los sistemas que tratan problemas de síntesis encuentran más dificultades en la validación. Así, validar un determinado plan generalmente suele implicar la actuación directa sobre el dominio de aplicación para ver su evolución, ya que no es fácil que existan estándares para probar su validez. Todo esto se complica si el dominio es crítico.

Relación con el entorno

Generalmente los sistemas expertos no actúan de forma separada, sino que se encuentran integrados en un sistema mayor (por ejemplo, el sistema de información de un hospital). En tal caso la validación de los interfaces con los otros elementos que forman el entorno (bases de datos, sistemas de entrada y salida, etc.) es fundamental para el correcto funcionamiento del sistema. Si el sistema no se integra en uno mayor y actúa por solitario su validación se hace más sencilla, al no ser necesario tener en cuenta su relación con el entorno.

7.1.3. Influencia de la fase de desarrollo

La fase en la que se encuentre el desarrollo del sistema también influirá en la forma de realizar la validación. Al describir la ingeniería del conocimiento habíamos comprobado que la metodología que mejor se adaptaba al desarrollo de sistemas expertos era una metodología incremental, en la que el sistema se iba construyendo en base a una serie de fases, en cada una de las cuales se añadían nuevas funcionalidades o se refinaban las ya existentes. Después de la terminación de una de estas fases se obtenía un prototipo que tenía que ser verificado y validado.

Cuanto más avanzado esté el desarrollo del sistema más compleja será la verificación y validación del mismo. Así, en las primeras etapas la validación consistirá en simples pruebas con casos ya resueltos, realizadas en un ambiente de laboratorio y que sólo cubrirán aspectos concretos del sistema. Posteriormente, el número de casos analizados por el sistema será mayor y también lo será su cobertura, la validación no será llevada a cabo sólo por el ingeniero del conocimiento sino que entrarán a formar parte de la misma expertos ajenos al desarrollo del sistema. Por último, en las fases finales primarán más los aspectos orientados al uso y se preparará el terreno para una validación en el entorno de trabajo por parte de los usuarios finales.

7.2. Fase de aplicación

La salida de la fase de planificación la constituyen una serie de estrategias que nos indican cómo realizar la validación del sistema experto que estamos considerando. Por ejemplo, entre estas estrategias podríamos tener:

- La validación sólo puede llevarse a cabo a través de una validación contra el experto, comparando los resultados de nuestro sistema con los resultados de un grupo de expertos.
- Como los usuarios son expertos y el dominio no es crítico pueden llevarse a cabo pruebas de campo.
- Como las interpretaciones del sistema siguen una escala ordinal es necesario ponderar las discrepancias según su distancia. Podemos utilizar medidas de pares como la kappa ponderada, el porcentaje de acuerdo dentro de uno o medidas de asociación.
- Las medidas de pares pueden utilizarse como entrada para los tests de grupo que permiten ordenar a los expertos humanos y al sistema experto según la similitud de sus diagnósticos.
- Al no existir una referencia estándar no es posible utilizar medidas como los ratios de acuerdo.
- etc.

La fase de aplicación será la encargada de llevar a la práctica estas recomendaciones obteniendo unos resultados que luego puedan ser interpretados en la siguiente fase.

La fase de aplicación se compone de los siguientes pasos (Figura 7.4):

(1) Captura de la casuística,

(2) Preprocesado de los datos, y

(3) Realización de las medidas estadísticas.

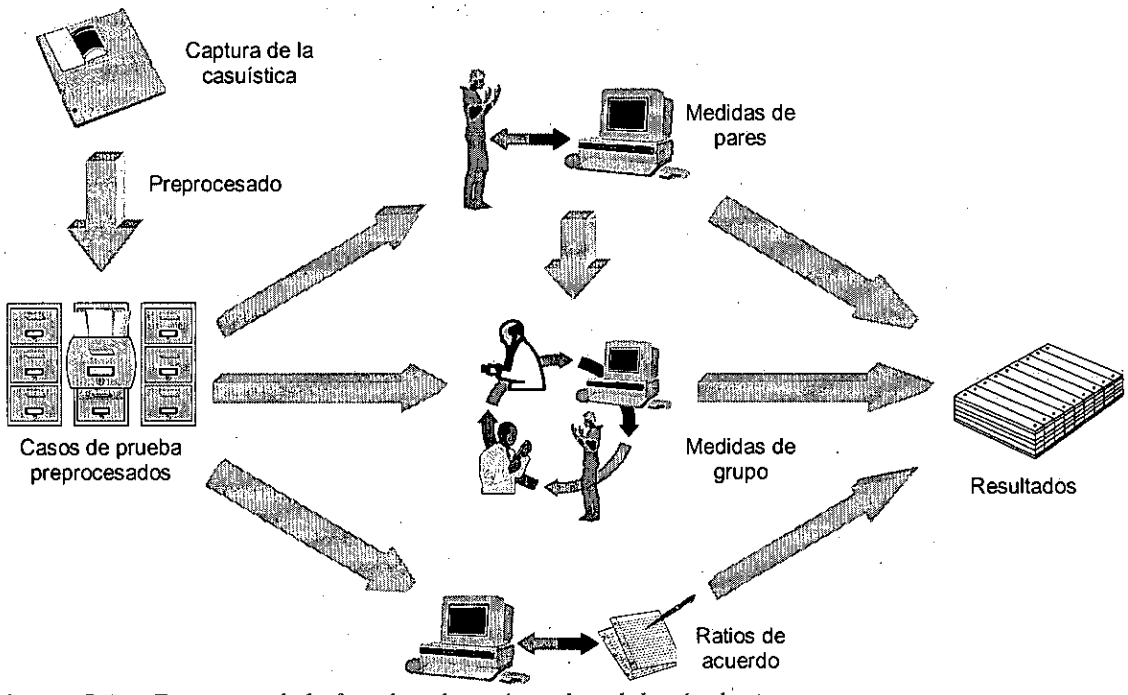


Figura 7.4. Estructura de la fase de aplicación en la validación de sistemas expertos.

7.2.1. Captura de la casuística

Para realizar la validación es necesario contar con una serie de casos ya resueltos con los que podamos comparar los resultados de nuestro sistema experto. Como indicábamos en el apartado 5.3 es necesario que la muestra obtenida cumpla dos características fundamentales: cantidad y representatividad. La muestra deberá ser suficientemente amplia y debe cubrir todos los aspectos que pretendemos probar en nuestro sistema.

La utilización de casos de prueba en la validación se adapta perfectamente a los métodos de desarrollo incremental, aumentando el número de casos y su cobertura a medida que avanzamos en el desarrollo..

7.2.2. Preprocesado de los datos

Una vez capturada la casuística tenemos una base de datos en la que se comparan los resultados de una o varias referencias (expertos humanos o la solución real del problema) con los resultados de nuestro sistema experto.

Sin embargo la base de datos obtenida, generalmente, no se puede utilizar directamente por los procesos de validación y es necesario llevar a cabo un preprocesado que puede incluir:

- *Corrección de errores*. Detectar y corregir los errores que presente la base de datos original (por ejemplo, categorías similares representadas por etiquetas distintas).

- *Transformación de los datos*. La forma en que se han recogido los datos puede no ser la más adecuada para llevar a cabo los procesos de validación y puede

ser necesario llevar a cabo una transformación de los mismos (por ejemplo transformar etiquetas lingüísticas en valores numéricos).

- *Inclusión de información adicional.* Muchas veces es necesario incluir nueva información en la base de datos original. Entre la diversa información que es necesario incluir destacamos las siguientes (que veremos más en detalle al describir la herramienta SHIVA):

  * *Estructura de la base de datos*: La base de datos de validación es una matriz tridimensional cuyas dimensiones son expertos, diagnósticos y casos. Sin embargo, los modelos de bases de datos actuales son tablas bidimensionales, por lo que podemos encontrarnos con diversas posibilidades al reducir los datos tridimensionales a una tabla bidimensional. Es necesario especificar cuál es la disposición de nuestra base de datos.
  * *Identificación de los campos de la base de datos*: Después de identificar la estructura de la base de datos es necesario identificar los tipos de sus campos (expertos, diagnósticos, casos, etc.).
  * *Orden de las categorías semánticas*: si los resultados forman parte de una escala ordinal es necesario especificar su orden.
  * *Ponderación de las discrepancias obtenidas*: las discrepancias encontradas entre las diversas categorías semánticas pueden variar en su importancia, por ello es necesario incluir algún tipo de ponderación que refleje estas diferencias.

Famili et al. (1996) destacan la importancia de la correcta selección de las técnicas de preprocesado para encontrar información útil durante el análisis.

7.2.3. Realización de medidas estadísticas.

Las estrategias de validación obtenidas en el proceso de planificación indican qué medidas estadísticas son las más adecuadas a utilizar en base a las características del dominio, del sistema y de la fase de planificación.

Las estrategias de validación también pueden sugerir el ámbito cualitativo en el que realizar estas medidas (análisis de sensibilidad, validación de subsistemas, tests de campo, pruebas de Turing, etc.).

Las medidas estadísticas incluidas en esta metodología son de tres tipos:

1. *Medidas de pares.* Se encargan de evaluar el grado de acuerdo o asociación entre expertos.

2. *Medidas de grupo.* Agrupan a los expertos según la similitud de sus diagnósticos utilizando como base los resultados de las medidas de pares.

3. *Ratios de acuerdo.* Comparan los resultados del sistema con una referencia estándar.

Estas medidas ya han sido ampliamente descritas en el punto anterior, sin embargo, en este apartado comentaremos algunos aspectos de las mismas que están íntimamente relacionados con su utilización en entornos de validación.

7.2.3.1. El porcentaje de acuerdo dentro de uno para el análisis de tendencias

La medida del porcentaje de acuerdo puede utilizarse en la validación para analizar tendencias. Para ilustrar esto podemos ver el trabajo de (Alonso et al., 1992) sobre el sistema experto NST-EXPERT.

En este caso en concreto se pretende analizar la capacidad del sistema experto y de varios expertos humanos a la hora de pronosticar la salida fetal final. El pronóstico se divide en 6 categorías: bueno (B), ligeramente malo (LM), moderadamente malo (MM), bastante malo (BM), muy posible malo (PM) y casi seguro malo (SM).

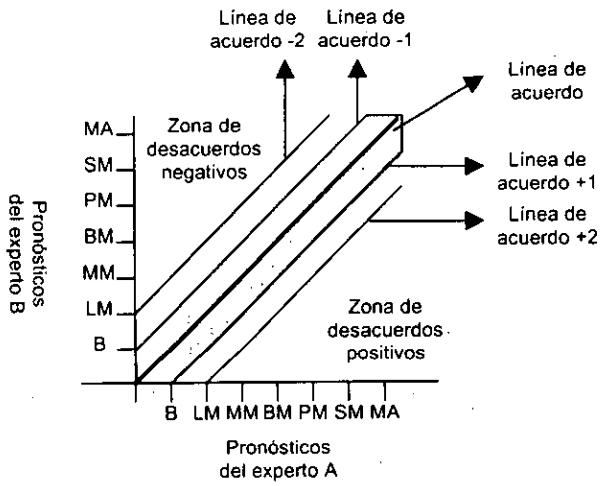


Figura 7.5. Análisis de tendencias con el porcentaje de acuerdo dentro de uno.

La zona de desacuerdos negativos indica que las predicciones realizadas por el experto B son más pesimistas que las del experto A, la zona de desacuerdos positivos indica que las predicciones realizadas por el experto B son más optimistas que las realizadas por el experto A. De esta forma, analizando los valores de acuerdo en las diagonales secundarias podemos ver si las interpretaciones de los expertos tienden hacia determinadas categorías.

7.2.3.2. Relación entre el porcentaje de acuerdo y kappa

Una situación interesante que puede suceder al utilizar el índice kappa en la validación es la siguiente: supongamos que tenemos las matrices de contingencia representadas en la Tabla 7.4 y en la Tabla 7.5.

		Experto B			
		BAJO	NORMAL	ALTO	
Experto A	BAJO	0.83 (0.818)	0.10	0.00	0.93
	NORMAL	0.05	0.00 (0.005)	0.00	0.05
	ALTO	0.00	0.00	0.02 (0.004)	0.02
		0.88	0.10	0.02	1

Tabla 7.4.

		Experto B			
		BAJO	NORMAL	ALTO	
Experto A	BAJO	0.28 (0.125)	0.10	0.00	0.38
	NORMAL	0.05	0.28 (0.125)	0.00	0.33
	ALTO	0.00	0.00	0.29 (0.084)	0.29
		0.33	0.38	0.29	1

Tabla 7.5.

Los valores del porcentaje de acuerdo y de kappa para cada una de ellas son los siguientes:

Tabla 7.4
$p_o = 0.85 = 85\ \%$
$p_c = 0.827$
$\kappa = 0.133$

Tabla 7.5
$p_o = 0.85 = 85\ \%$
$p_c = 0.335$
$\kappa = 0.774$

Como vemos, en ambas tablas el porcentaje de acuerdo es el mismo (85%), sin embargo el valor de kappa es muy inferior en la primera tabla mientras que es un valor alto en la segunda.

En la Tabla 7.4 el valor de kappa es bajo porque el porcentaje de acuerdo debido a la casualidad es alto, y este porcentaje es alto porque la dispersión de los datos en esta matriz es muy escasa (si nos fijamos podemos ver como la mayoría de los casos pertenecen a la celda "BAJO"-"BAJO"). Sin embargo, en la Tabla 7.5 los casos se distribuyen más uniformemente entre las distintas celdas y el porcentaje de acuerdo debido a la casualidad no es tan elevado (0.335) lo que permite que el valor de kappa sea mayor.

Este comportamiento de kappa ha motivado las críticas de diversos autores como (Donker et al., 1992) y (Gjørup, 1988) en las que señalan que el valor de kappa está muy influido por la distribución de las probabilidades marginales y que, para poder interpretar un valor de kappa, sería necesario acompañarlo de su correspondiente matriz de contingencia.

Sin embargo, este comportamiento de kappa es normal (incluso deseable) si atendemos a su definición: "una medida de acuerdo que corrige aquellos acuerdos debidos a la casualidad". Desde luego, si la dispersión de los casos es escasa la

probabilidad de acertar por casualidad aumenta y, consecuentemente, el valor de kappa disminuye.

7.2.3.3. Kappa ponderada y status de los expertos

En los ejemplos vistos hasta ahora de kappa ponderada, se asignaban pesos iguales a celdas simétricas ($v_{ij} = v_{ji}$), lo que quiere decir que el desacuerdo que se produce cuando el experto A decide el diagnóstico #1 y el experto B el diagnóstico #2 es igual al producido cuando la situación es a la inversa. Esto es apropiado cuando estamos intentando comprobar la consistencia entre dos fuentes de datos tienen el mismo *status*. Si las fuentes de datos tienen diferente *status* (una actúa como predictor y la otra como criterio), puede ser razonable que celdas simétricas tengan distintos pesos ($v_{ij} \neq v_{ji}$).

Así podemos tener un experto A, que es el sistema experto que queremos validar y un experto B que es el criterio que se considera correcto según un consenso de especialistas en la materia. Entonces se puede considerar que un error del sistema al dar el diagnóstico #2 cuando el criterio consensuado dice que el diagnóstico correcto es el #3 es más grave que si el hecho sucede a la inversa. Para que el índice kappa ponderada considere esta situación se deben asignar los pesos de forma apropiada, tal y como se muestra en la Tabla 7.6. (como vemos el peso de la casilla (2, 3) es 6 mientras que el peso de la casilla (3, 2) es sólo 2).

		Consenso		
		#1	#2	#3
Sistema Experto	#1	0	1	4
	#2	1	0	6
	#3	2	2	0

Tabla 7.6. Pesos utilizados con expertos que tienen diferente status.

7.2.3.4. Tau en tablas de contingencia

En el capítulo anterior describíamos como obtener la medida tau a partir de una muestra de datos. Los datos de validación generalmente son muestras grandes que se agrupan en tablas de contingencia, por lo que se necesita un método para calcular tau a partir de estas tablas.

Por ejemplo, supongamos la tabla de contingencia que relacionaba los resultados de los expertos A y SE (Tabla 6.3.) que reproducimos a continuación y en la que a cada categoría semántica se le ha asignado un determinado rango.

			Experto SE					
			MUY BAJO (1)	BAJO (2)	NORMAL (3)	ALTO (4)	MUY ALTO (5)	
Experto A	MUY BAJO	(1)	0	1	0	0	0	1
	BAJO	(2)	0	2	1	0	0	3
	NORMAL	(3)	0	0	3	0	0	3
	ALTO	(4)	0	1	0	1	0	2
	MUY ALTO	(5)	0	0	0	0	1	1
			0	4	4	1	1	10

El valor de una casilla (i, j) en la tabla anterior representa el número de casos a los que el experto A ha asignado el rango i y el experto SE ha asignado el rango j. Nuestro primer paso para calcular τ debe ser calcular los índices C y D, por ejemplo los pares concordantes con los de la casilla (1, 1) serían los pares (2, 2), (2, 3), ..., (5, 5) mientras que no habría pares discordantes. Sin embargo, como la casilla (1, 1) está vacía no contribuye de ninguna forma a los índices C y D.

Consideremos entonces los tres valores de la casilla (3, 3). Los pares concordantes con esta celda serían (4, 4), (4, 5), (5, 4) y (5, 5); multiplicando los tres valores de la casilla (3, 3) con la suma de valores de las casillas concordantes nos da que la contribución de esta casilla al índice C es $3\times(1+1) = 6$ (no se cuentan como concordantes los pares (1, 1), (1, 2), (2, 1) y (2, 2) porque ya han sido considerados con anterioridad). Los pares discordantes con (3, 3) son (4, 1), (4, 2), (5, 1) y (5, 2). La contribución de la casilla (3, 3) al índice D es $3\times(1) = 3$ (los pares (1, 4), (1, 5), (2, 4) y (2, 5) no se cuentan como discordantes porque ya han sido considerados con anterioridad). Procediendo de esta manera con todas las celdas que tengan valores no nulos obtenemos los índices C y D (Tabla 7.7).

Pares	Contribución a *C*			Contribución a *D*		
(1, 2)	1 × (1+3+1+1)	=	6	1 × (0)	=	0
(2, 2)	2 × (3+1+1)	=	10	2 × (0)	=	0
(2, 3)	1 × (1+1)	=	2	1 × (1)	=	1
(3, 3)	3 × (1+1)	=	6	3 × (1)	=	3
(4, 2)	1 × (1)	=	1	1 × (0)	=	0
(4, 4)	1 × (1)	=	1	1 × (0)	=	0
(5, 5)	1 × (0)	=	0	1 × (0)	=	0
			26			***4***

Tabla 7.7. Cálculo de los índices C y D a partir de la tabla de contingencia.

De esta forma vemos que $C = 26$ y que $D = 4$, lo que nos permite calcular τ como (26-4)/45 = 0.489. Sin embargo siempre que tratamos con tablas de contingencia de validación es normal que el número de ligaduras sea elevado (ya que, por lo general, el número de casos disponibles siempre será mayor que el número de casillas en la tabla de contingencia) por ello una medida más adecuada de la asociación sería la τ_b de Kendall. Para el cálculo de τ_b es necesario calcular los valores U y V de la tabla de contingencia, para ello se procede de la siguiente forma: considerando que todos los valores que están en una misma fila o columna están ligado, los marginales de las filas y de las columnas representan respectivamente el número de valores existentes en la ligadura de esa fila o columna, esto nos permite hallar los valores de U y V según la equ. 6.30 y la equ. 6.31.

u (marginales de A)	*u (u - 1)*	*v* (marginales de SE)	*v (v - 1)*
1	0	0	0
3	3	4	6
3	3	4	6
2	1	1	0
1	0	1	0
	7		***12***

Tabla 7.8. Cálculo de las ligaduras a partir de la tabla de contingencia.

Por lo tanto $U = 7$ y $V = 12$, lo que implica que el valor de τ_b es

$$\tau_b = \frac{26-4}{\sqrt{(45-7)(45-12)}} = \frac{22}{35.412} = 0.621$$

Existe otra revisión del índice τ (denominada τ_c) pero sólo es utilizada en tablas de contingencia en las que el número de columnas es distinto al número de filas. Como las tablas de contingencia que relacionan las interpretaciones de los expertos siempre tienen el mismo número de filas y columnas, se decidió no incluir a τ_c en nuestro estudio.

En base a los valores de C y D también podemos calcular el valor de la gamma de Goodman-Kruskal como

$$\gamma = \frac{C-D}{C+D} = \frac{26-4}{26+4} = \frac{22}{30} = 0.733$$

7.2.3.5. Rho en tablas de contingencia

De la misma forma que el índice Tau, es necesario que Rho pueda ser hallado a partir de tablas de contingencia para simplificar su cálculo en entornos de validación. Si suponemos la misma tabla de contingencia del ejemplo anterior (la Tabla 6.3. que relaciona los resultados del experto A y del SSEE), el primer paso sería el cálculo de los rangos.

Para calcular los rangos debemos basarnos en la información de los datos marginales. Así, atendiendo a los marginales de A en la casilla (1, .) tenemos un único valor que equivale a el rango 1, en la casilla (2, .) tenemos 3 valores, a los que les pertenecerían los rangos 2, 3, y 4; por lo que el rango medio es 3. Procediendo de la misma forma con los demás marginales obtenemos:

			Experto SE						
			MUY BAJO (1)	BAJO (2)	NORMAL (3)	ALTO (4)	MUY ALTO (5)		Rangos II
Experto A	MUY BAJO	(1)	0	1	0	0	0	1	1
	BAJO	(2)	0	2	1	0	0	3	3
	NORMAL	(3)	0	0	3	0	0	3	6
	ALTO	(4)	0	1	0	1	0	2	8.5
	MUY ALTO	(5)	0	0	0	0	1	1	10
			0	4	4	1	1	10	
	Rangos	=	0	2.5	6.5	9	10		

Tabla 7.9. Cálculo de los rangos para los resultados de los expertos A y SE.

Una vez calculados los rangos podemos calcular la contribución de cada celda al índice d_i^2. Por ejemplo, en la celda (3, 3) tenemos tres valores que relacionan a los rangos 6 y 6.5. La contribución de esta celda a d_i^2 será $3 \times (6\text{-}6.5)^2 = 3 \times 0.25 = 0.75$. Actuando de la misma forma para aquellas celdas de valores no nulos obtenemos:

Celdas	Frecuencia	Rango A	Rango SE	Contribución a d_i^2 $F \times (R_A - R_{SE})^2$
(1, 2)	1	1.0	2.5	2.25
(2, 2)	2	3.0	2.5	0.50
(2, 3)	1	3.0	6.5	12.25
(3, 3)	3	6.0	6.5	0.75
(4, 2)	1	8.5	2.5	36.00
(4, 4)	1	8.5	9.0	0.25
(5, 5)	1	10.0	10.0	0.00
				52.00

Tabla 7.10. Contribuciones a d_i^2 obtenidas a partir de la tabla de contingencia que relaciona los resultados de los expertos A y SE.

Ahora es necesario corregir el efecto de las ligaduras, para ello no tenemos más que emplear los valores marginales de la tabla de la siguiente forma:

u (marginales de A)	u^3	v (marginales de SE)	v^3
1	1	0	0
3	27	4	64
3	27	4	64
2	8	1	1
1	1	1	1
10	***64***	***10***	***130***
	u' = (64-10) / 12 = 4.5		**v' = (130-10) / 12 = 10**

Tabla 7.11. Cálculo de las ligaduras a partir de la tabla de contingencia.

Una vez hallados Σd_i^2, u' y v' podemos hallar r_s según la equ. 6.39 de la siguiente forma:

$$r_s = \frac{1000 - 10 - 6(52) - 6(14.5)}{\sqrt{[1000 - 10 - 12(4.5)][1000 - 10 - 12(10)]}} == \frac{591}{\sqrt{(936)(870)}} = .655$$

7.2.3.6. Relación entre tau, rho y r de Pearson en entornos de validación.

La τ de Kendall, la r_s de Spearman y la r de Pearson pueden considerarse casos particulares de un coeficiente de correlación generalizado (Γ). Este coeficiente se define de la siguiente forma:

$$\Gamma = \frac{\sum_{i=1,\, j=1}^{n} a_{ij} b_{ij}}{\sqrt{\sum_{i=1,\, j=1}^{n} a_{ij}^2 \sum_{i=1,\, j=1}^{n} b_{ij}^2}}$$

equ. 7.1

en donde a_{ij} es un índice definido para los valores de x que debe cumplir que $a_{ij} = -a_{ji}$. De la misma forma b_{ij} es un índice definido para los valores de y que debe cumplir que $b_{ij} = -b_{ji}$. También se cumple que $a_{ij} = b_{ij} = 0$ si $i = j$.

Según la definición que adoptemos para a_{ij} y b_{ij}, podemos obtener el coeficiente τ, r_s o r. Así, si denotamos como p_i el rango del objeto i para los valores de x y q_i el rango del objeto i para los valores de y, obtenemos:

Valor de Γ	*Función a_{ij}*	*Función b_{ij}*
τ	$a_{ij} = \begin{cases} +1 & \text{si } p_i < p_j \\ -1 & \text{si } p_i > p_j \end{cases}$	$b_{ij} = \begin{cases} +1 & \text{si } q_i < q_j \\ -1 & \text{si } q_i > q_j \end{cases}$
r_s	$a_{ij} = p_j - p_i$	$b_{ij} = q_j - q_i$
r	$a_{ij} = x_j - x_i$	$b_{ij} = y_j - y_i$

Tabla 7.12. Valores de a_{ij} y b_{ij} para obtener τ, r_s y r a partir de Γ.

Como vemos el cálculo de τ es el más sencillo ya que asigna una unidad (positiva o negativa) a un par de casos según sean concordantes o no, pero sin tener en cuenta la separación existente entre los rangos. Sin embargo, r_s representa un cálculo más elaborado ya que da más peso a aquellos casos que estén más separados y su diferencia entre rangos es mayor. El caso de r es similar al de r_s pero atendiendo a los valores de la variables y no a los rangos que representan su ordenación. La demostración de los resultados de la Tabla 7.12 puede encontrarse en (Kendall y Gibbons, 1990).

Los tests basados en la tau de Kendall y la rho de Spearman son alternativas no paramétricas a los clásicos tests de independencia basados en el coeficiente de correlación de Pearson (que requiere que los datos sean medidos al menos en una escala intervalo y que la distribución sea la normal bivariante). Los tests no paramétricos requieren que los datos sean medidos al menos en una escala ordinal y la asunción de una distribución bivariante continua, no necesariamente la normal.

Los valores de tau y rho pueden relacionarse a través de desigualdades, entre las que podemos destacar la desigualdad de Daniels (1950):

$$-1 \leq 3\tau - 2r_s \leq 1$$

equ. 7.2

Muchos autores prefieren el uso de tau, a pesar que rho puede resultar más familiar por sus similitudes con el coeficiente de correlación de Pearson. Dos son las razones que motivan estas preferencias. Primero, la tau de Kendall tiene una interpretación sencilla y específica como la proporción de pares concordantes en la muestra menos la proporción de pares discordantes (que puede ser vista como la diferencia entre la probabilidad de que los datos observados estén en el mismo orden menos la probabilidad de que estén en ordenes diferentes). La rho de Spearman no tiene una interpretación tan intuitiva (es la correlación lineal de los rangos obtenidos de los valores de las variables). Segundo, La distribución muestral de rho tiende a la distribución normal de forma más lenta que la distribución muestral de tau.

Sin embargo, para el caso de la validación de expertos resultan más adecuadas las características de rho que las de tau. En primer lugar porque, como ya hemos dicho, rho tiene en cuenta la distancia entre las categorías mientras que tau sólo tiene en cuenta que al pasar de un caso a otro, las categorías designadas por los expertos varíen en una misma dirección (un salto entre la categoría ↑↑ y la categoría ↓↓, es tratado de la misma forma que un salto entre ↑↑ y ↑).

En segundo lugar tenemos el problema de los empates. Al tratar con tablas de contingencia los empates suelen muy numerosos. Con tau tenemos tres formas de tratarlos: ignorarlos en el numerador (τ), ignorarlos en el numerador corrigiendo el denominador (τ_b) e ignorarlos en numerador y denominador (γ). Puede que ninguna de las opciones sea satisfactoria para el caso de la validación. Supongamos así los siguientes pares de valores:

A	B
↑	↑↑
=	↓

(a)

A	B
↑	↑
↑	=

(b)

A	B
↑	↑
↑	↑

(c)

Tabla 7.13. Pares de valores para tau.

Los dos casos de la Tabla 7.13a son concordantes, ya que los resultados de ambos expertos disminuyen de valor, sin embargo el salto del experto A es menor cuantitativamente que el del experto B. En la Tabla 7.13b vemos como los resultados del experto A están ligados, sin embargo estos casos representan un desacuerdo ya que el salto de categoría en el experto B no ha su correspondencia en el experto A. También se podría argumentar que la diferencia de salto es menor que la producida en el caso (a) considerado concordante. En la Tabla 7.13c vemos como de nuevo tenemos una ligadura, aunque los valores representen un acuerdo perfecto.

El tratamiento de las ligaduras en rho parece más comprensible dentro del contexto de la validación de expertos. Así supongamos que tenemos los siguientes valores:

Casos	A	B
1	↑	↑
2	↑↑	=
3	=	↓↓
4	↑	↓

(a)

Casos	A	B
1	2.5	4
2	4	3
3	1	1
4	2.5	2

(b)

Tabla 7.14. Valores para rho.

Al pasar los valores a rangos vemos que se ha producido una ligadura en el experto A (que hemos solucionado aplicando rangos medios). Un hecho que puede llamar la atención y puede parecer un error es que el aparente acuerdo entre A y B en el primer caso (↑,↑) se ha traducido en rangos poco concordantes (2.5, 4). Esto tiene su explicación si analizamos la distribución de los casos de A y B (Figura 7.6).

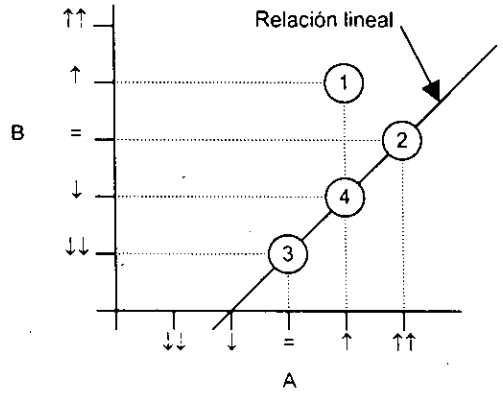


Figura 7.6. Distribución de los casos de la Tabla 7.14. Los números en los círculos indican el número de cada caso.

Como vemos los diagnósticos de los expertos parecen tener una clara relación lineal en la cual el experto B selecciona siempre dos categorías por debajo de la categoría seleccionada por A. Este hecho puede darse por una mala definición de las categorías que lleva a los expertos a interpretarlas de manera diferente. En tal situación el caso uno con categorías (↑,↑) no sería una asociación válida porque en el contexto general se separa de la relación lineal existente entre A y B.

Un resumen de las ventajas e inconvenientes de las distintas medidas de asociación puede consultarse en la Tabla 7.15.

Medida	Ventajas	Inconvenientes
r de Pearson	Conocida y ampliamente utilizada	Asunción de una distribución normal bivariante
	Adecuada cuando lo importante son los valores y no el orden de las categorías	No adecuada a datos ordinales, puede variar ante transformaciones que mantienen el orden
r de Kendall	Medida no paramétrica	No adecuada cuando lo importante son los valores y no el orden de las categorías
	Adecuada a datos ordinales, es invariable ante transformaciones que mantienen el orden	No tiene en cuenta el grado de separación entre las categorías
	Interpretación sencilla y específica	Manejo de las ligaduras confuso en contextos de validación
	Converge a la normal más rápido que rho	
r_s de Spearman	Medida no paramétrica	No adecuada cuando lo importante son los valores y no el orden de las categorías
	Adecuada a datos ordinales, es invariable ante transformaciones que mantienen el orden	Interpretación no tan sencilla como la de tau
	Tiene en cuenta el grado de separación de las categorías	Converge a la normal más lento que tau
	Manejo de las ligaduras de forma más coherente en contextos de validación	

Tabla 7.15. Ventajas e inconvenientes de las distintas medidas de asociación.

7.2.3.7. Medidas de Williams

La medida de Williams nos ofrece un método eficaz para comprobar si el nivel de acuerdo o asociación de nuestro sistema experto con los expertos humanos, es similar al nivel de acuerdo o asociación existente entre los propios expertos humanos.

Sin embargo, los resultados de esta medida pueden ser fácilmente mal interpretados si dentro del grupo de referencia existe un experto claramente en desacuerdo con los demás, o si el acuerdo dentro del grupo de referencia es escaso. Motivos de esta falta de acuerdo ya se habían detallado en el capítulo 5.4.1 en el que se incluyen: factores externos, influencias de expertos de mayor nivel, ambigüedades en los datos, pertenencia a distintas escuelas de pensamiento, etc.

Esta falta de acuerdo entre los expertos provocará que el valor de P_n (acuerdo entre los expertos del grupo) sea bajo y, por lo tanto, valores bajos de acuerdo entre el experto aislado y el grupo (P_0) generen valores de I_n cercanos a la unidad.

En tales casos se deberá estudiar si el experto en desacuerdo con los demás debería ser apartado del grupo de referencia o tratar de utilizar técnicas para el desarrollo un consenso dentro del grupo.

7.2.3.8. Análisis cluster

En el capítulo anterior veíamos las principales características del análisis cluster, y en concreto, del análisis cluster jerárquico. En este apartado trataremos de detallar las

principales características del análisis cluster cuando es aplicado en entornos de validación.

Selección de las variables relevantes

En la validación de sistemas expertos, la selección de las variables relevantes es muy sencilla: la matriz de elementos estaría formada por los n expertos que llevan a cabo la validación y por los d casos sobre los cuales los expertos realizan sus diagnósticos. De esta forma, la matriz de elementos $n \times d$ (en este caso n son las columnas y d las filas), es idéntica a la base de datos de validación que ya hemos visto al describir los tests de pares (Tabla 6.2) y que mostramos a continuación.

Casos	A	B	C	D	SE
1	ALTO	ALTO	ALTO	ALTO	ALTO
2	ALTO	BAJO	ALTO	ALTO	BAJO
3	BAJO	BAJO	NORMAL	NORMAL	BAJO
4	NORMAL	BAJO	NORMAL	NORMAL	NORMAL
5	MUY ALTO	MUY ALTO	ALTO	ALTO	MUY ALTO
6	BAJO	BAJO	BAJO	BAJO	NORMAL
7	MUY BAJO	MUY BAJO	NORMAL	BAJO	BAJO
8	NORMAL	NORMAL	NORMAL	ALTO	NORMAL
9	NORMAL	NORMAL	BAJO	MUY BAJO	NORMAL
10	BAJO	BAJO	BAJO	ALTO	BAJO

Selección de las medidas de similitud

Las principales medidas de similitud o disimilitud utilizadas son medidas métricas, sin embargo, tal y como explican Aldenderfer y Blashfield (1984), las características del problema a tratar pueden indicar la utilización de otro tipo de medidas que no tengan porque se necesariamente métricas. Este es el caso de la matriz de elementos utilizada en la validación de sistemas expertos que veíamos en la Tabla 6.2. En este caso tenemos una serie de expertos que realizan opiniones sobre una serie de casos. Los casos constituyen las variables de nuestro estudio y generalmente ocurrirá que el número de casos (variables) es mucho mayor que el número de expertos.

Los mejores coeficientes de similitud que podemos definir sobre estos datos son las medidas de pares que veíamos en el capítulo 6.1. Con ellas podemos medir acuerdos y asociaciones teniendo en cuenta pesos entre las distintas categorías, acuerdos debidos a la casualidad, etc. Estas medidas de pares cumplen algunas características que habíamos nombrado antes (simetría, están limitadas entre [0, 1] o [-1, 1], etc.) pero generalmente no pueden considerarse métricas.

Así, ateniéndonos a los casos de la Tabla 6.2, la matriz de similitudes para el porcentaje de acuerdo será idéntica a la tabla resumen utilizada en las medidas de Williams (Tabla 7.16).

	A	B	C	D	SE
A	—	0.8	0.6	0.4	0.7
B	0.8	—	0.4	0.2	0.7
C	0.6	0.4	—	0.6	0.4
D	0.4	0.2	0.6	—	0.3
SE	0.7	0.7	0.4	0.3	—

Tabla 7.16. Matriz de similitudes para el porcentaje de acuerdo.

Tipo de análisis cluster

Dentro de los distintos tipos posibles de análisis cluster, el más adecuado para aplicar en la validación de sistemas expertos es el conocido con las siglas SAHN, que corresponden a un análisis cluster secuencial, aglomerativo, jerárquico y sin superposición. Los principales motivos para utilizar este tipo de clustering son: (1) necesita sólo la matriz de similitudes para poder ser aplicado, lo que permite agrupar expertos utilizando como medidas de similitud las medidas de pares, y (2) no es necesario indicar el número de clusters a obtener porque el mismo algoritmo nos detalla una solución que incluye todos los posibles números de cluster.

Dentro de este tipo de algoritmos encontramos varias diferencias a la hora de establecer las distancias entre los nuevos clusters creados. De los distintos métodos vistos en el capítulo anterior no son aplicables a la validación el del centroide, el de la mediana y el de Ward porque sólo pueden calcularse directamente de la matriz de distancias si ésta esta formada por distancias euclídeas al cuadrado. Los restantes métodos (distancias mínimas, distancias máximas, promedio entre grupos y media) son aplicables pero se suele preferir el método del promedio entre grupos porque incorpora información de todos los individuos que forma el cluster, y porque las distancias obtenidas no son ambiguas.

7.2.3.9. Relación entre el MDS y el análisis cluster jerárquico.

Tanto el análisis cluster como el MDS persiguen un mismo objetivo, la representación de los expertos según sus grados de acuerdo, pero mientras el análisis cluster lo realiza mediante una serie de uniones anidadas el MDS lo hace mediante una representación gráfica en 2D.

El análisis cluster se concentra en representar de forma más exacta las similitudes mayores. Así en las primeras etapas del proceso las similitudes extraídas del dendrograma son "cercanas" a las similitudes originales. Sin embargo, a medida que los clusters crecen en tamaño las similitudes del cluster son menos comparables con las similitudes originales.

Supongamos de nuevo los resultados para el porcentaje de acuerdo entre los expertos A, B, C, D y SE. Las similitudes originales son:

	A	B	C	D	SE
A	0	0.2	0.4	0.6	0.3
B	0.2	0	0.6	0.8	0.3
C	0.4	0.6	0	0.4	0.6
D	0.6	0.8	0.4	0	0.7
SE	0.3	0.3	0.6	0.7	0

La realización de un análisis cluster jerárquico nos permite obtener el siguiente dendrograma:

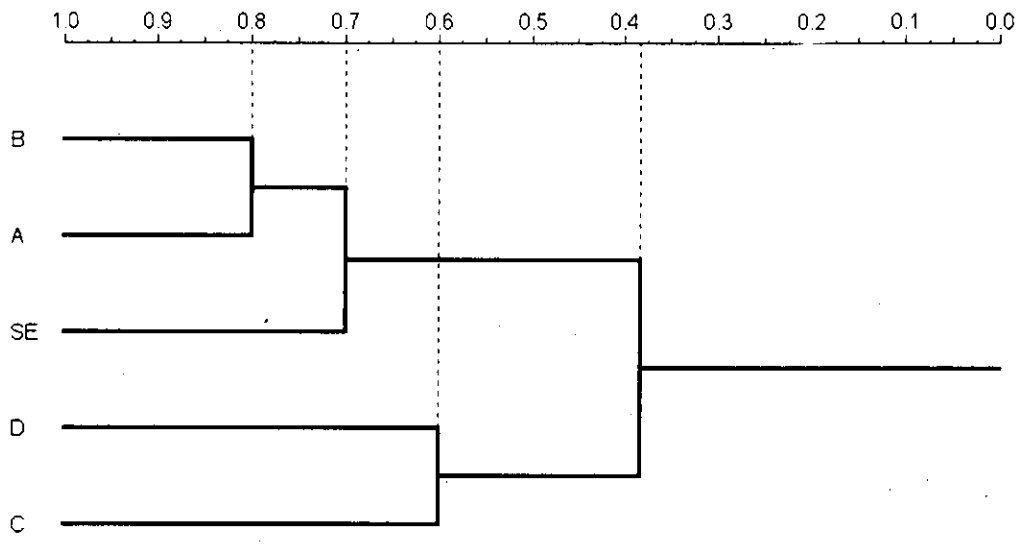


Figura 7.7. Dendrograma del análisis cluster para los datos del porcentaje de acuerdo utilizando el algoritmo del promedio entre grupos.

Como vemos las primeras uniones representan a la perfección las similitudes iniciales: A-B = 0.8, A-SE = 0.7, B-SE = 0.7 y C-D = 0.6. Sin embargo la última unión del dendrograma entre los clusters A.B.SE y C.D no representa tan fielmente dichas similitudes iniciales (A-C originalmente era 0.6 y ahora es 0.383).

Por otro lado, los resultados del MDS tienden a actuar de forma opuesta a como lo hace el análisis cluster, es decir, a representar de forma más exacta las similitudes más pequeñas cometiendo errores mayores en las similitudes más grandes.

Por ejemplo, si convertimos en similitudes las distancias euclídeas obtenidas al realizar un análisis MDS a los datos del porcentaje de acuerdo el resultado sería:

	A	B	C	D	SE
A	0	0.792	0.618	0.399	0.729
B	0.792	0	0.417	0.199	0.739
C	0.618	0.417	0	0.625	0.400
D	0.399	0.199	0.625	0	0.315
SE	0.729	0.739	0.400	0.315	0

Si ordenamos los pares según las diferencias encontradas entre estas similitudes y las similitudes originales obtenemos el siguiente resultado:

Pares	Diferencia
C – SE	0.000
B – D	0.001
A – D	0.001
A – B	0.008
D – SE	0.015
B – C	0.017
A – C	0.018
C – D	0.025
A – SE	0.029
B – SE	0.039

Como vemos los pares que presentan menores diferencias son aquellos correspondientes a los expertos más lejanos, mientras que las mayores diferencias se dan en los expertos más cercanos.

Generalmente, el mejor indicador de cual de las dos soluciones (cluster o MDS) es la que representa más fielmente los datos originales, es la medida de correlación entre las distancias originales y las resultantes de la aplicación del método. En este caso la correlación para el análisis cluster (correlación cofenética) es de 0.864 mientras que la correlación para el MDS es de 0.998. Evidentemente la mejor solución es la del MDS.

Muchas veces suele ser muy útil combinar los resultados del análisis cluster con los resultados del MDS. Esto se realiza mediante la gráfica de burbujas ya vista en la Figura 6.25 en la que a la representación 2D del MDS se le superpone los resultados del análisis cluster. En el caso del porcentaje de acuerdo la gráfica de burbujas sería tal y como se representa en la Figura 7.8.

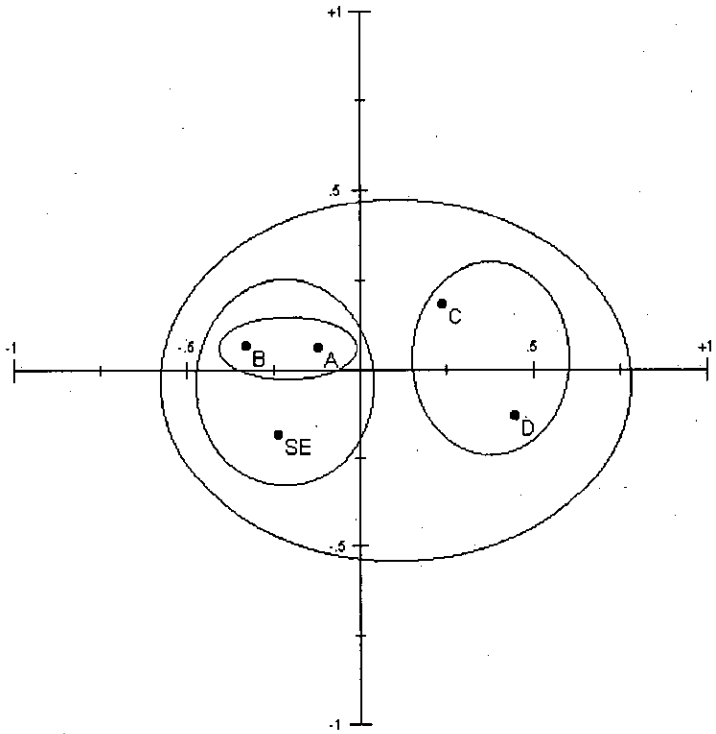


Figura 7.8. Solución del MDS métrico para los datos del porcentaje de acuerdo con la superposición del análisis cluster en forma de gráfico de burbujas.

7.2.3.10. Relación entre el MDS y el análisis factorial

La técnica del análisis factorial presenta muchas similitudes con el escalamiento multidimensional. El objetivo del análisis factorial es: (1) reducir el número de variables y (2) detectar estructuras en las relaciones entre las distintas variables.

A pesar de las similitudes, el análisis factorial y el escalamiento multidimensional son dos método diferentes. El análisis factorial requiere que los datos subyacentes se distribuyan siguiendo una normal multivariable, y que las relaciones sean lineales. El MDS no impone esas restricciones.

También el análisis factorial tiende a extraer más factores (dimensiones) que el MDS, lo que implica que los resultados del MDS son soluciones más fáciles de interpretar. Por último, el MDS puede ser aplicado a cualquier tipo de distancias o similitudes, mientras que el análisis factorial obtiene las similitudes a partir de una matriz de correlaciones.

7.2.3.11. Medida de Jaccard

La medida de Jaccard es muy útil para medir acuerdos en aquellas situaciones en las que resultan más importantes los resultados positivos que los negativos. Por ejemplo, supongamos la categoría diagnóstica ALTO y su correspondiente matriz de acuerdo que vimos en la Tabla 6.48 y que reproducimos a continuación.

		Referencia Estándar	
		ALTO	¬ALTO
Sistema Experto	ALTO	1	0
	¬ALTO	1	8
		2	8

Porcentaje = 0.9
Jaccard = 0.5

Vemos que el porcentaje de acuerdo es casi perfecto (0.9) pero este porcentaje se debe fundamentalmente a que la mayoría de los casos caen en la casilla (¬ALTO, ¬ALTO). En el coeficiente Jaccard, los resultados de esta casilla no se tienen en cuenta y el nivel de acuerdo baja hasta el 0.5. En situaciones donde el número de categorías es elevado y la probabilidad de caer en la casilla ($\neg D$, $\neg D$) es elevada, el coeficiente Jaccard es más recomendable que el porcentaje de acuerdo.

Un ejemplo típico de la utilización de la medida de Jaccard aparece cuando queremos medir la similitud entre especies de animales. Evidentemente un camello y un pez carecen ambos de alas, pero esto no indica mucho a favor de su similitud. Lo más correcto sería fijarse sólo en aquellas características que comparten.

7.3. Fase de interpretación

La fase de interpretación se basa en los resultados de la fase de aplicación para dilucidar si el sistema experto se comporta realmente como un experto dentro de su campo de aplicación. Su esquema puede verse en la Figura 7.9

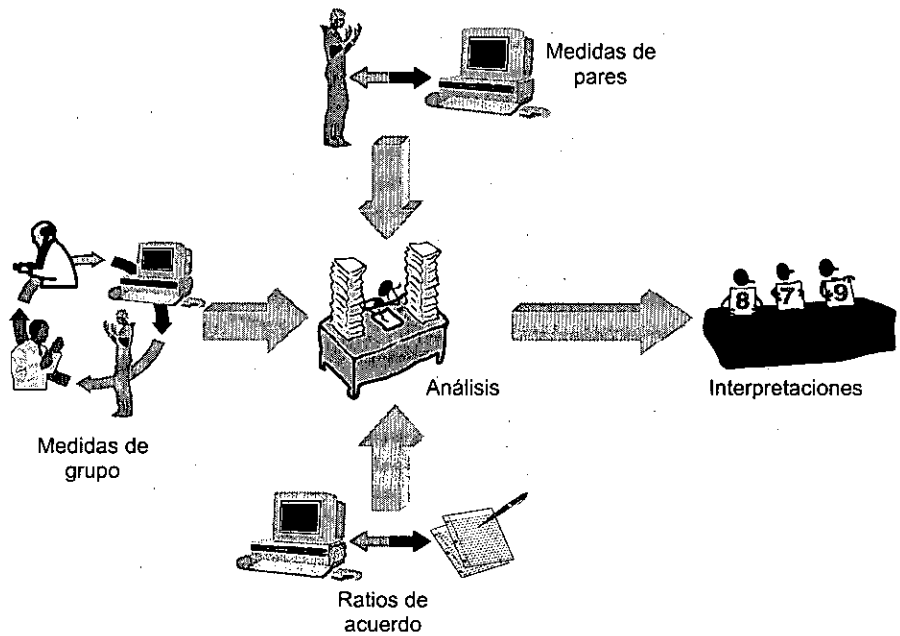


Figura 7.9. Estructura de la fase de interpretación en la validación de sistemas expertos.

La fase de interpretación es quizás la más compleja de la metodología. Esto es así porque los resultados de los tests estadísticos dependen mucho de la naturaleza del problema que estamos tratando, y de la muestra empleada en su obtención.

Además, puede ser muy complicado obtener una simple respuesta de SI o NO a la validez del sistema debido, generalmente, a la complejidad del problema tratado. O'Keefe et al. (1987) señalan que lo importante en el proceso de validación es comprobar que el rendimiento del sistema alcanza unos límites que se consideren aceptables (ya que la validez absoluta es casi imposible de conseguir). La interpretación de los resultados se complica también por el hecho de que unos resultados que son aceptables para un determinado domino, pueden ser completamente inaceptables para otros (porque, por ejemplo, se trate de dominios críticos).

El proceso de interpretación también puede ser utilizado para adquirir nuevo conocimiento o para refinar el conocimiento ya existente.

Tanto la fase de planificación como la de interpretación son fases heurísticas en las que se intentará modelar la experiencia del ingeniero de conocimiento a la hora de validar los sistemas inteligentes. Por otro lado, la fase de aplicación es eminentemente algorítmica.

7.4. Resumen

Después de haber descrito los principales paradigmas que caracterizan el proceso de validación, es necesario el desarrollo de una metodología formal que tenga en cuenta dichos paradigmas. La metodología de validación que hemos propuesto consta de tres fases: planificación, aplicación e interpretación.

La fase de planificación se encarga de definir las estrategias a seguir en el proceso de validación en base a las características del dominio, las características del propio sistema y las características de la fase de desarrollo en la que nos encontremos.

La fase de aplicación usa como base las estrategias obtenidas de la fase de planificación, para llevar a cabo una serie de pruebas cuantitativas, que pueden ser aplicadas en un entorno cualitativo. La fase de aplicación se divide a su vez en tres subfases: captura de la casuística, preprocesado de los datos y realización de las medidas estadísticas (entre las que incluimos medidas de pares, medidas de grupo y ratios de acuerdo).

Por último, la fase de interpretación analiza los resultados de las medidas obtenidos en la fase anterior para tratar de dilucidar si el sistema experto se comporta realmente como un experto dentro de su dominio de aplicación. Esta fase también puede ser empleada para adquirir conocimiento o refinar el conocimiento ya existente.

Para facilitar la aplicación de las distintas fases de la metodología hemos desarrollado una herramienta informática (SHIVA) que describiremos en el siguiente capítulo.

8. LA HERRAMIENTA DE VALIDACIÓN "SHIVA"

> Si la única herramienta que tienes es un martillo, tiendes a ver todos los problemas como si fueran clavos.
> *Abraham Maslow (Psicólogo estadounidense, 1908 – 1970).*
>
> Los hombres se han convertido en las herramientas de sus herramientas.
> *Henry David Thoreau (Escritor, filósofo y naturalista estadounidense, 1817 – 1862).*

Para facilitar la aplicación de la metodología de validación propuesta en el capítulo anterior, se hacía necesaria la construcción de una herramienta que automatizara sus fases.

La herramienta SHIVA (Sistema Heurístico e Integrado de Validación) acepta como entrada una base de datos (BD) de validación, como la vista en la tabla Tabla 6.2, y permite seguir los pasos marcados por la metodología de validación descrita en el apartado 7 de planificación, aplicación (incluyendo preprocesado de datos y realización de medidas de pares, medidas de grupo y ratios de acuerdo), para facilitar al usuario la última fase de interpretación.

8.1. Características de la implementación

La herramienta SHIVA ha sido desarrollada en la utilidad de programación Borland Delphi 3.0 que utiliza como lenguaje base una versión de Pascal orientada a objetos (Object Pascal). La elección de esta herramienta se ha basado en características como la facilidad de uso, rapidez, fiabilidad, etc., que la hacen superior a otras herramientas de programación como Visual Basic o Visual C++ en diversos campos. (el lenguaje Pascal es más sencillo que el C++ y casi tan potente, además el código de Delphi es compilado y no interpretado, como el del Basic, con las consiguientes ventajas en velocidad de ejecución).

La herramienta SHIVA consta de un total de 28 módulos que suman unas 14,000 líneas de código. La estructura de los módulos principales de SHIVA se muestra en la Figura 8.1.

El módulo *ExpertPlan* contiene las rutinas necesarias para la ejecución de un pequeño sistema experto que permite realizar, de forma sencilla, las tareas de planificación de la validación. Aunque el sistema experto ha sido desarrollado en la herramienta *Nexpert Object*, la implementación de un pequeño motor de inferencias en el módulo *Ssee* permite ejecutarlo de forma independiente a la herramienta, ganando así en rapidez y configurabilidad. Este pequeño motor de inferencias también es utilizado por el sistema experto de ayuda a la interpretación del módulo *ExpertInt*.

Para facilitar el preprocesado de la base de datos inicial se ha construido un asistente que se compone de tres fases (incluidas en los módulos *As_Val_1*, *As_Val_2* y *As_Val_3*). El módulo *IO_Val* contiene las rutinas necesarias para la transferencia de los datos de memoria a disco.

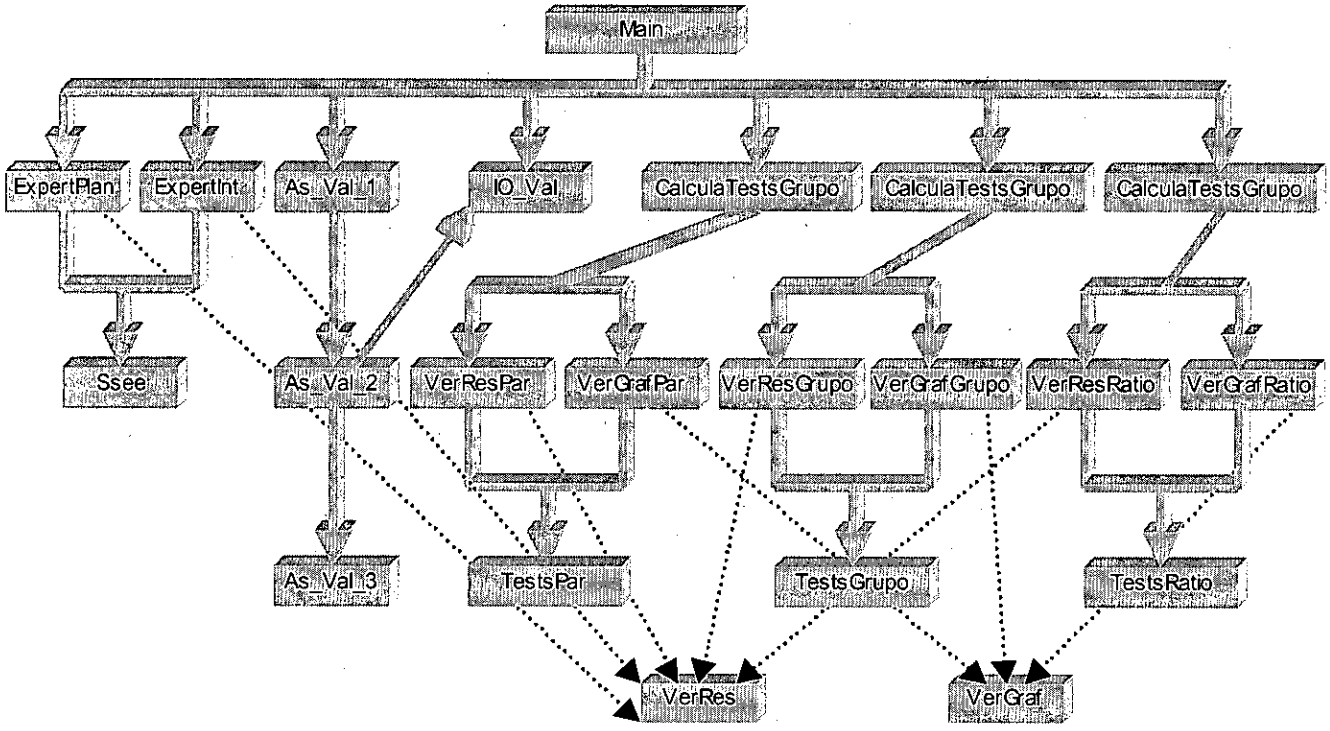


Figura 8.1. Estructura de los módulos principales de SHIVA.

Los módulos *CalculaTestsPar*, *CalculaTestsGrupo* y *CalculaTestsRatio* se ocupan, respectivamente, de la realización de las medidas de pares, las medidas de grupo y los ratios de acuerdo. Cada uno de ellos utiliza módulos separados para mostrar los resultados de forma gráfica (*VerGrafPar*, *VerGrafGrupo* y *VerGrafRatio*), o en modo texto (*VerResPar*, *VerResGrupo* y *VerResRatio*). Los algoritmos matemáticos de las medidas se hallan incluidos en los módulos *TestsPar*, *TestsGrupo* y *TestsRatios*, mientras que los visores gráficos y de texto utilizados por todas las medidas se hallan incluidos en *VerGraf* y *VerRes* respectivamente (*VerRes* también se utiliza para mostrar los resultados de los sistemas expertos de planificación e interpretación).

Además de estos módulos existen otros que, bien por ser usados por muchos de los módulos principales, o por tener escasa entidad, no se han incluido en el gráfico. Estos módulos son:

- *About*: Contiene la versión del programa e información sobre los autores.
- *AlmaDat*: Especifica la estructura dinámica encargada de almacenar los datos en memoria.
- *Constant*: Contiene constantes, tipos, variables y procedimientos globales al resto de módulos del sistema.
- *EscribeF*: Contiene rutinas que facilitan la escritura de texto, tanto en el módulo *VerRes* como en ficheros de texto.
- *MnjVal*: Maneja la información adicional introducida en el proceso de preprocesado.
- *Options*: Guarda las opciones más comunes empleadas en SHIVA en el fichero SHIVA.INI.

8.2. Ventana principal

La ventana principal de SHIVA nos permite acceder a todas las opciones disponibles en la herramienta. Dicha ventana se representa en la Figura 8.2 y está formada por 6 zonas diferenciadas: Menú de opciones, Barra de opciones, Características de la BD, Barra de progreso, Sistemas expertos y Tests de validación.

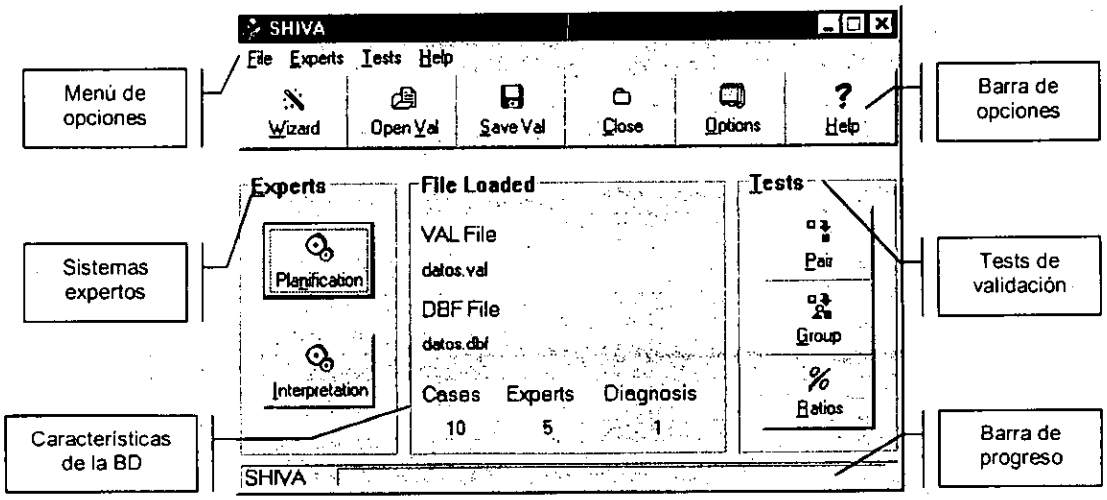


Figura 8.2. Menú principal de la herramienta SHIVA

El menú de opciones permite escoger las distintas opciones del programa a través de menús desplegables. Sin embargo, para hacer más fácil la utilización de SHIVA, la mayoría de las opciones del menú son accesibles desde botones en la ventana principal.

La barra de opciones incluye los botones para preprocesar una nueva base de datos, abrir una base de datos ya preprocesada, guardar una base de datos, cerrar una base de datos, cambiar las opciones más comunes del programa y acceder al fichero de ayuda. Una vez que se halla abierto una base de datos sus características se muestran en la parte central de la ventana principal. Estas características incluyen los nombres de los ficheros de validación (cuyo formato describiremos en el apartado 8.4.1) además de los casos de validación, expertos y diagnósticos utilizados en el estudio. La barra de progreso se activa al abrir una base de datos y muestra el porcentaje de la misma que ya ha sido leído.

En la parte izquierda de la ventana se incluyen los elementos heurísticos de la herramienta, esto es, el experto de planificación y el experto de interpretación. En la parte derecha se incluyen los elementos determinísticos, es decir, las medidas de pares, las medidas de grupo y los ratios de acuerdo.

A continuación describiremos como se implementan en SHIVA las distintas fases de la metodología de validación propuesta.

8.3. Sistema experto de planificación

Como habíamos comentado al describir la metodología, existen muchos aspectos distintivos en el proceso de validación. La elección de una técnica concreta de validación vendrá determinada por las características del dominio de aplicación, las

características del propio sistema y las características de la fase de desarrollo en la que nos encontremos. En la fase de planificación lo que se pretende es analizar dichas características y decidir qué técnicas de validación son las más adecuadas para nuestro sistema.

El módulo de planificación de la herramienta SHIVA tiene como objetivo facilitar el proceso de elección de las técnicas de validación mediante un pequeño sistema experto. La ventana que forma el interfaz con el sistema experto aparece representados en la Figura 8.3. En ella podemos ver que la planificación de la validación se hará en base a las características del dominio de aplicación, las características del sistema, y la fase de desarrollo en que nos encontremos.

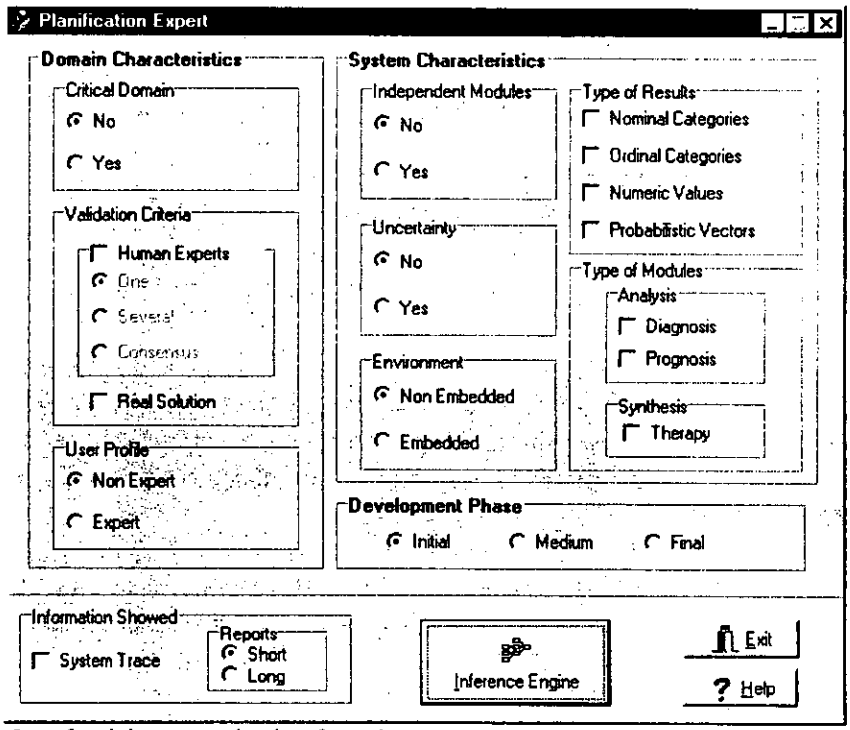


Figura 8.3. Interfaz del experto de planificación.

8.3.1. Descripción del sistema

El sistema experto de planificación ha sido desarrollado en la herramienta *Nexpert Object*. Está formado por unas 50 reglas que se configuran en una estructura en árbol que puede verse en la Figura 8.4.

La dirección del motor de búsqueda es regresiva, partiendo de una única hipótesis "Check_Plan" que sirve como punto de partida de nuestro sistema experto. Así la primera regla de nuestro sistema tiene la siguiente estructura:

```
RULE :  Rule CHK_PLAN
If
        Check_Domain is TRUE
        And Check_System is TRUE
        And Check_Development is TRUE
Then Check_Plan
        is confirmed.
```

Al lanzar una búsqueda regresiva sobre "Check_Plan" el sistema lanza tres nuevas búsquedas regresivas: "Check_Domain" para analizar las características del

dominio, "Check_System" para analizar las características del sistema, y "Check_Development" para analizar las características de la fase de desarrollo.

Las reglas del sistema experto de planificación pueden consultarse en el apéndice A de esta memoria.

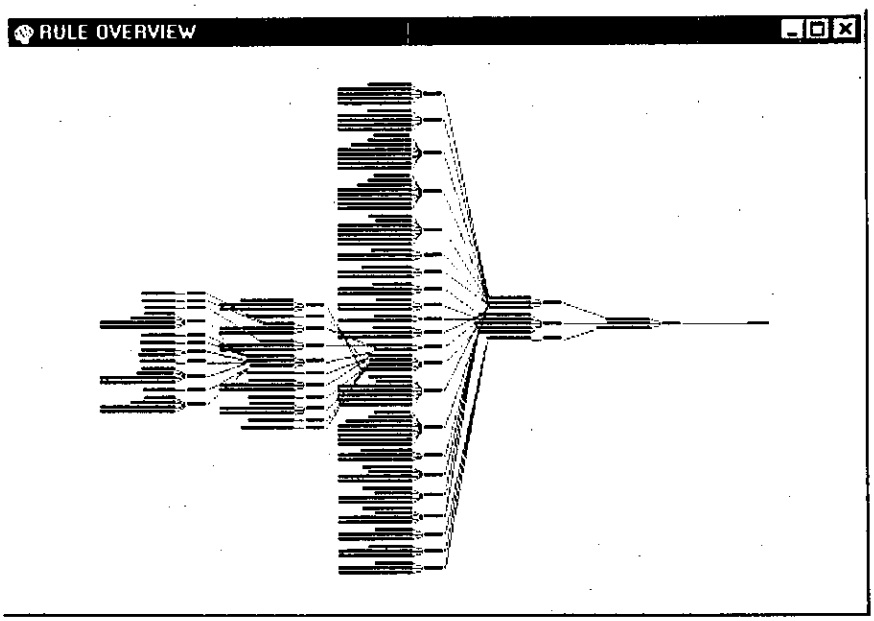


Figura 8.4. Esquema de las reglas del experto de planificación.

8.3.2. Motor de inferencias de SHIVA

Con el objetivo de que las consultas fueran más rápidas, y el sistema experto más fácil de configurar, se incluyó en la herramienta SHIVA un pequeño motor de inferencias que permitía llevar a cabo la búsqueda regresiva. También se hizo necesario desarrollar las estructuras dinámicas en memoria que se encargarían de alojar las reglas y los objetos del sistema. Todo este desarrollo se incluyó dentro del módulo *Ssee* al que tienen acceso, tanto el sistema experto de planificación, como el sistema experto de interpretación.

El organigrama del procedimiento que realiza la búsqueda regresiva se representa en la Figura 8.6 y ha sido adaptado de Castillo y Alvarez (1989). La leyenda para su interpretación puede verse en la Figura 8.5

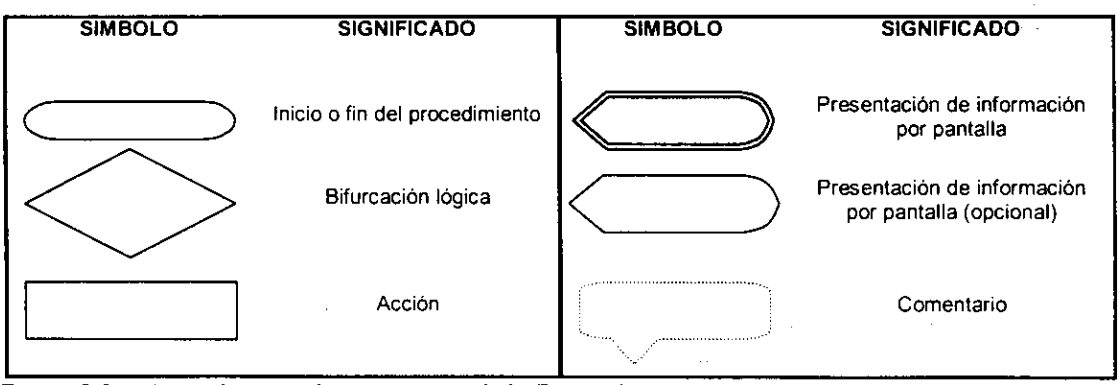


Figura 8.5. Leyenda para el organigrama de la Figura 8.6

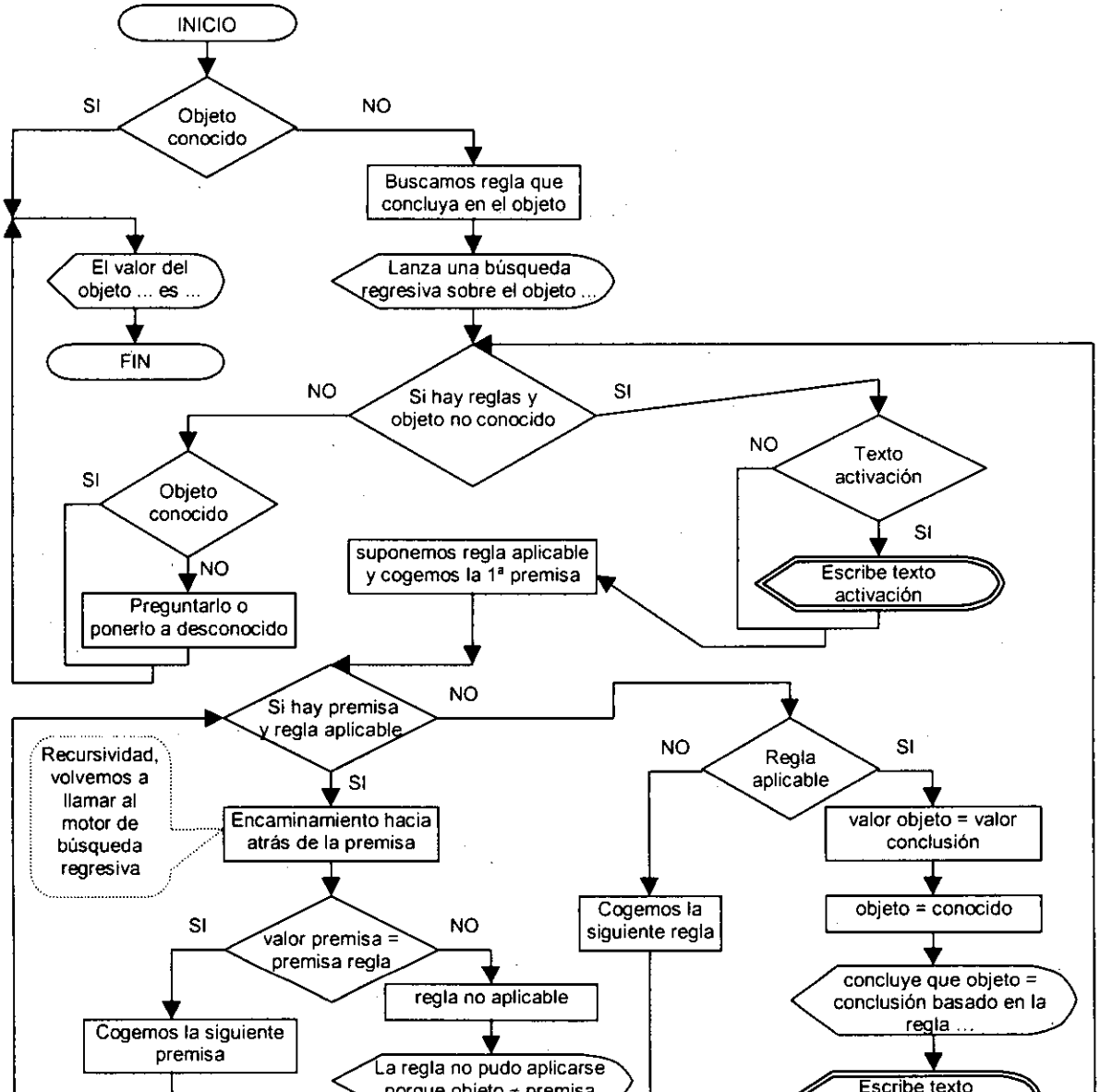


Figura 8.6. Organigrama del motor de búsqueda regresiva.

El algoritmo realiza una búsqueda regresiva para conocer el valor de un determinado objeto. El primer paso es preguntar si el valor de dicho objeto ya es conocido y, en caso afirmativo, devolverlo. En caso de que el valor del objeto sea desconocido es necesario buscar una regla que lo incluya en su conclusión.

El bucle principal del algoritmo consiste en ir revisando todas las reglas de la base de conocimientos, hasta que se pueda concluir el valor del objeto, o hasta que se hayan acabado todas las reglas. El análisis de una regla en particular se realiza de la siguiente forma: en primer lugar se analiza la primera premisa de la regla y se realiza una búsqueda regresiva para hallar su valor (volvemos a llamar recursivamente al algoritmo). Una vez que hemos encontrado el valor de la premisa, se compara con el valor que está especificado en la regla, y si son iguales pasamos a la siguiente premisa. Si no son iguales la regla no se puede aplicar.

Si todas las premisas de una regla son ciertas, la regla es aplicable con lo que al objeto que aparece en la conclusión se le puede asignar el valor indicado en la regla. En caso de que la regla no sea aplicable se busca otra que también concluya en el objeto a buscar.

Si hemos acabado todas las reglas y aún no hemos podido obtener el valor del objeto buscado nos quedan dos alternativas, preguntar el valor al usuario o poner su valor a *desconocido*.

Cada regla puede llevar asociado un texto de activación (que aparecerá cuando se active la regla) y un texto de conclusión (que aparecerá cuando se ejecuta). El resto de información por pantalla sólo aparece en caso de que se solicite, para facilitar así el seguimiento de los procesos realizados por el motor de inferencias.

8.3.3. Ejemplo de funcionamiento

Como ejemplo de funcionamiento del módulo de planificación consideremos un ejemplo con cuatro expertos humanos y el sistema experto como el que hemos visto hasta ahora, y supongamos una serie de características tanto del dominio como del sistema o la etapa de desarrollo.

En cuanto a las características del dominio, consideraremos que el sistema experto que estamos validando es un sistema experto médico que se va a ejecutar en un entorno crítico. No existe un estándar para la validación y, en su caso, utilizamos la información proveniente de cuatro expertos humanos. Los usuarios finales del sistema serán también expertos humanos.

En cuanto a las características del sistema, éste consta de un único módulo que establece el valor de un determinado diagnóstico. Este diagnóstico está dividido en cinco categorías que forman una escala ordinal. El sistema no incluye incertidumbre ni está embebido en un sistema mayor.

La etapa de desarrollo en la que nos encontramos es intermedia, es decir, el sistema está bastante evolucionado pero aún falta bastante para obtener la versión final.

En base a estos datos la pantalla de interfaz del sistema experto de planificación quedaría de la siguiente forma:

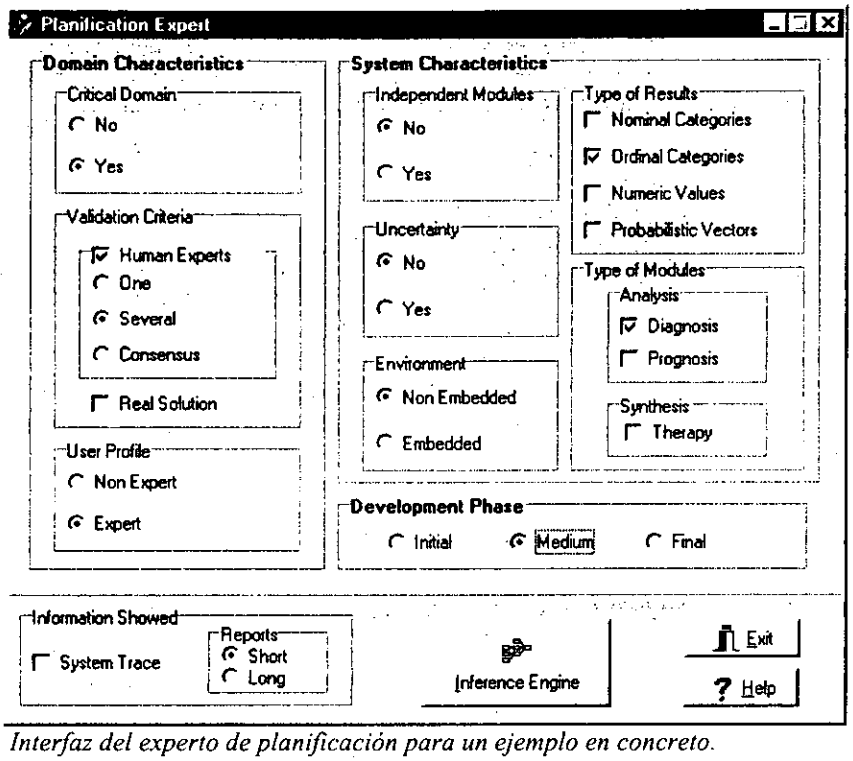


Figura 8.7. Interfaz del experto de planificación para un ejemplo en concreto.

Los resultados del sistema aparecen en la ventana que se representan en la Figura 8.8 en su versión corta, en la Figura 8.9 en su versión larga y en la Figura 8.10 mostrando la traza de ejecución del motor de inferencias.

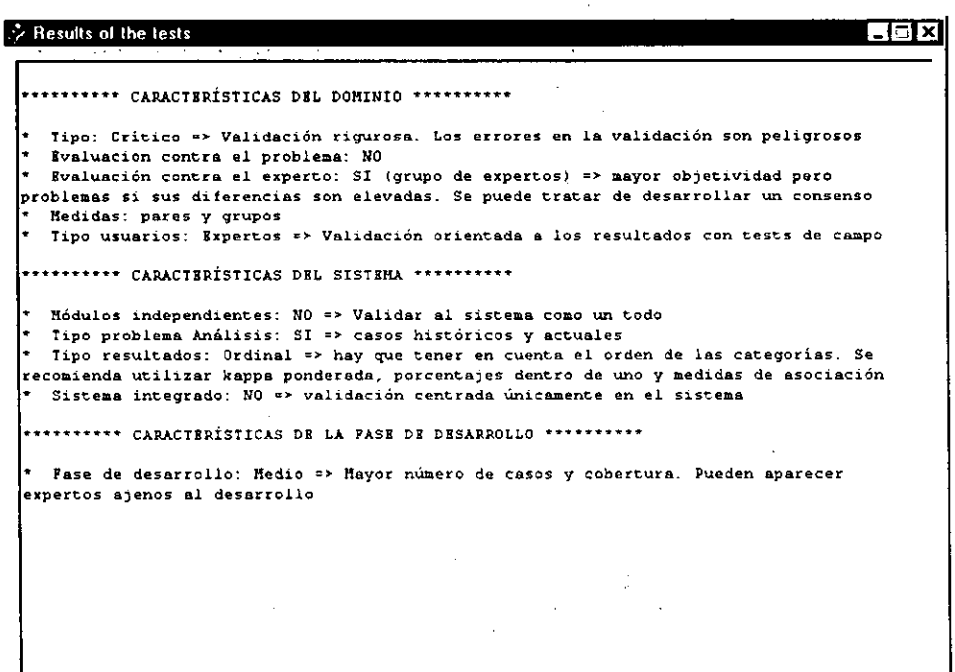


Figura 8.8. Ventana de resultados del sistema experto de planificación (versión corta).

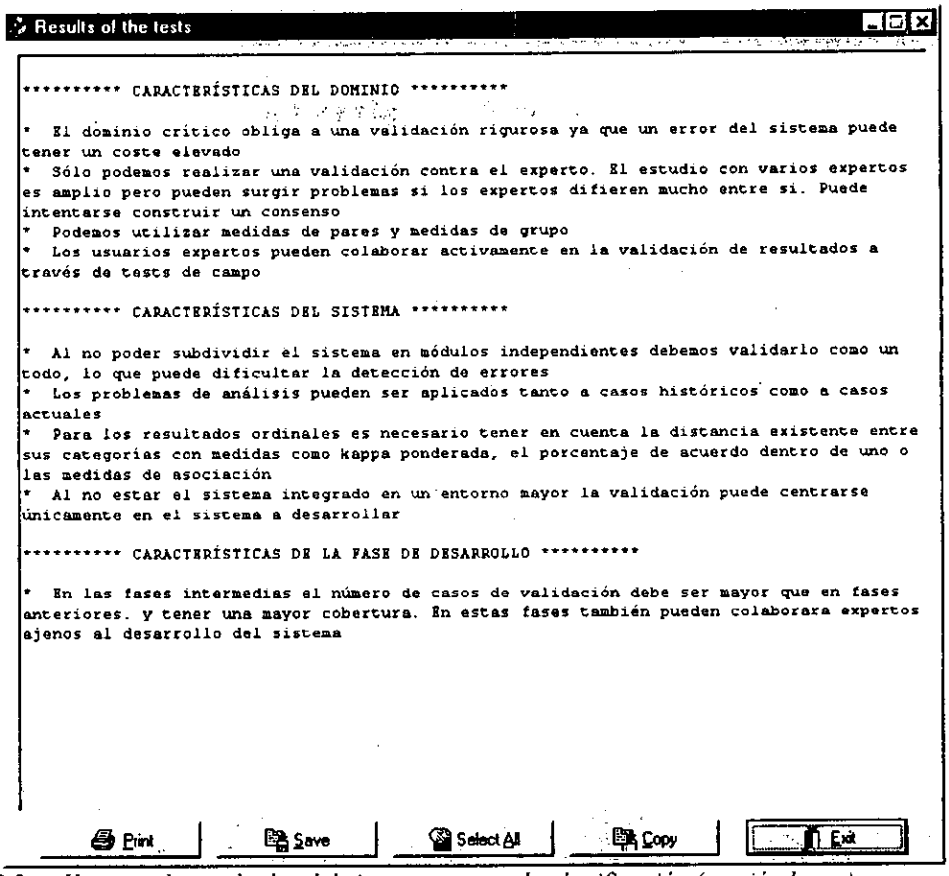


Figura 8.9. Ventana de resultados del sistema experto de planificación (versión larga).

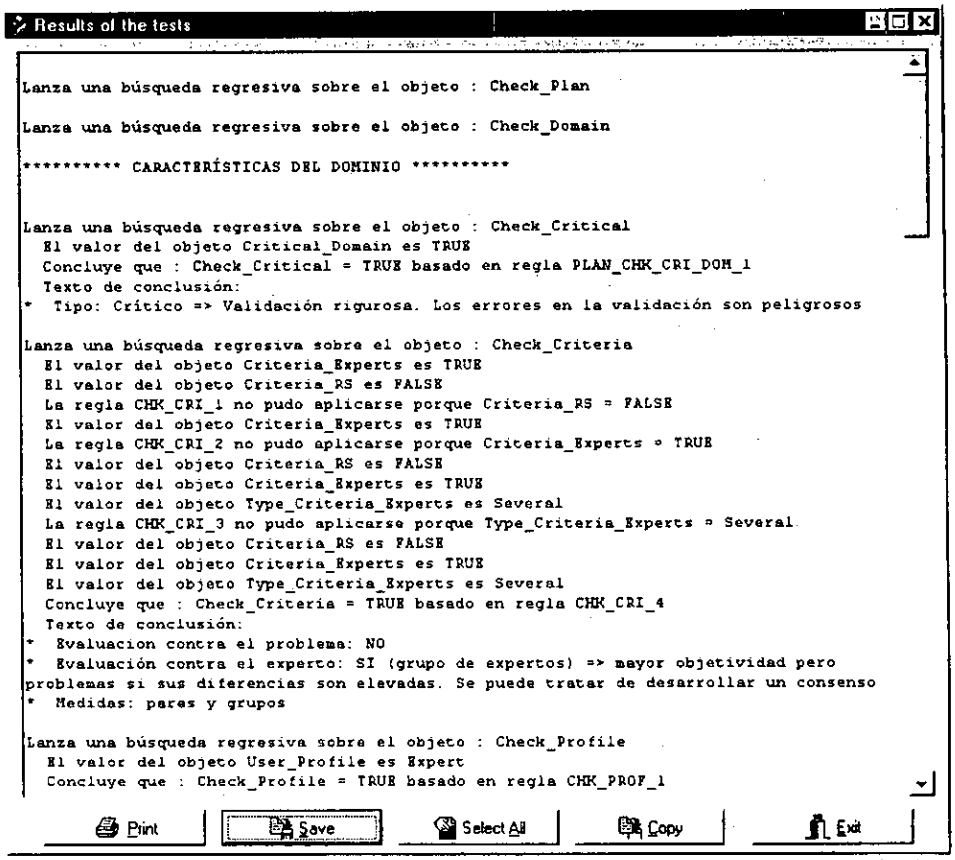


Figura 8.10. Ventana de resultados del sistema experto de planificación (mostrando la traza de ejecución del motor de inferencias).

8.4. Aplicación de las medidas de validación

La segunda fase de la metodología de validación se denomina *fase de aplicación* porque comprende la aplicación determinista de los algoritmos que permiten calcular las medidas de pares, las medidas de grupo y los ratios de acuerdo.

Un paso previo para el cálculo de estas medidas es la captura de la casuística de validación y su preprocesado.

8.4.1. Preprocesado de los datos de validación

Una vez hemos capturado los datos de validación es necesario preprocesarlos. Algunas de las tareas del preproceso deben hacerse de forma externa a SHIVA (como la corrección de errores o la transformación de los datos). Sin embargo para la tarea de inclusión de información adicional, la herramienta SHIVA provee un asistente que facilita dicho proceso. Esta tarea es necesario realizarla porque la información contenida en la base de datos de validación es insuficiente para llevar a cabo dicha validación.

SHIVA acepta como formato de base de datos de validación ficheros en el estándar dBase (DBF).

8.4.1.1. Formatos de las bases de datos

El primer paso del asistente es decidir cuál es el fichero DBF a utilizar y cómo se organizan los expertos y los diagnósticos en la base de datos. La base de datos de validación es una array tridimensional cuyas dimensiones son expertos, diagnósticos y casos (esta es la forma en que los datos se almacenan en memoria). Sin embargo los ficheros dBase son tablas bidimensionales, por lo que podemos encontrarnos con diversas posibilidades al reducir los datos tridimensionales a un fichero dBase.

La principal decisión que hay que hacer es dónde localizar a los expertos y a los diagnósticos. Existen dos posibilidades, colocarlos en filas o en columnas, lo que da lugar a cuatro configuraciones diferentes (Figura 8.11):

1) *Expertos en columnas y diagnósticos en filas (ECDF).* La configuración de la base de datos sería como muestra la Figura 8.11a. Los expertos aparecen como campos de la base de datos mientras que los diagnósticos aparecen en las filas, existiendo un solo campo de la base de datos que se encarga de identificar a qué diagnóstico pertenece cada caso. Suele utilizarse cuando el número de expertos es mayor que el número de diagnósticos.

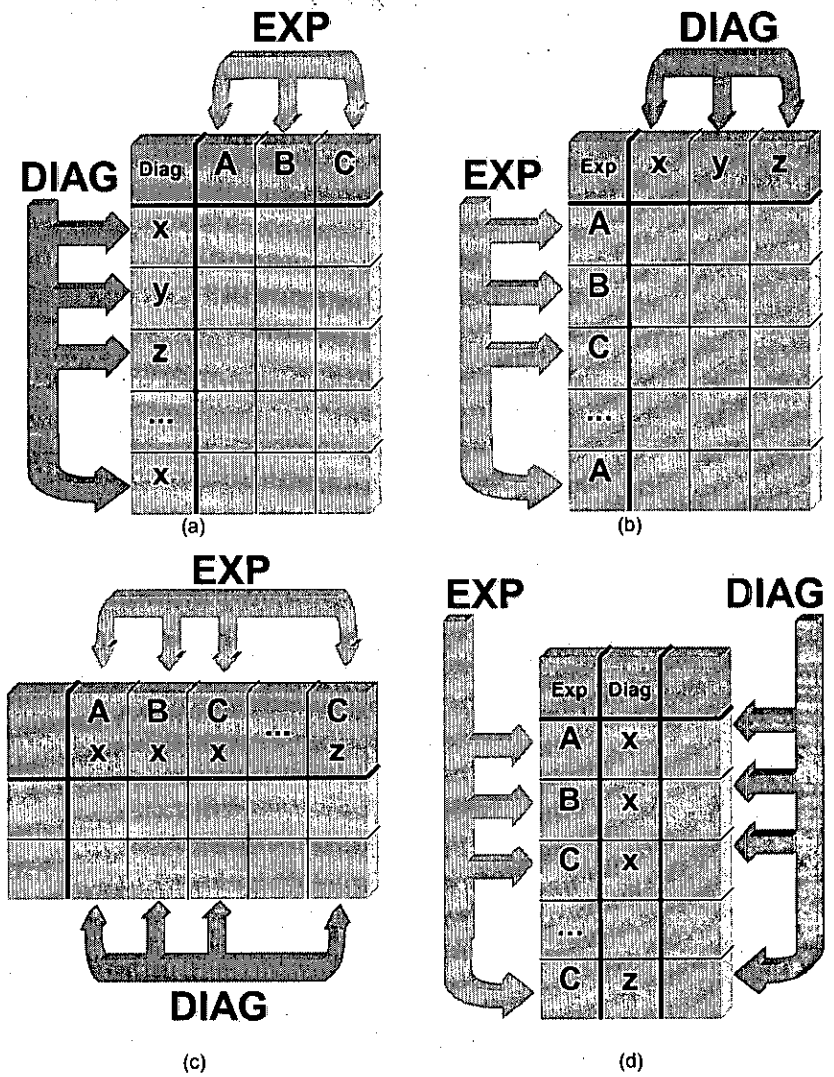


Figura 8.11. Cuatro posibles configuraciones de la base de datos: (a) ECDF, (b) EFDC, (c) ECDC y (d) EFDF.

2) *Expertos en filas y diagnósticos en columnas (EFDC)*: Es la configuración inversa a la anterior (Figura 8.11b). En este caso los campos de la base de datos son los diagnósticos y los expertos se distribuyen en filas existiendo un campo especial que identifica al experto de cada caso. Se utiliza normalmente cuando el número de diagnósticos es mayor que el número de expertos.

3) *Expertos en columnas y diagnósticos en columnas (ECDC)*: En este caso existe un campo en la base de datos por cada posible combinación de expertos y diagnósticos (Figura 8.11c). El número de campos de la base de datos puede ser muy elevado, pero toda la información de un caso se halla contenida en un único registro.

4) *Expertos en filas y diagnósticos en filas (EFDF)*: En esta configuración (Figura 8.11d) sólo existen tres campos en la base de datos, uno que indica el experto, otro que indica el diagnóstico y por último un tercero que indica el valor que el experto indicado ha seleccionado para ese diagnóstico. En este caso el número de filas de la base de datos puede ser muy elevado.

Las configuraciones más comunes que aparecen son ECDF y EFDC, es decir, distribuir expertos y diagnósticos en filas y columnas; no poner a ambos en la misma posición relativa.

Los casos ECDC y EFDF pueden fácilmente reducirse a los anteriores. Por ejemplo, podemos dividir la base de datos ECDC en varias bases de datos agrupando las columnas que comparten los mismos diagnósticos. En este caso tendremos tantas bases de datos del modelo ECDF como diagnósticos tengamos. Con un poco más de trabajo estas bases de datos ECDF pueden unirse en una sola (Figura 8.12).

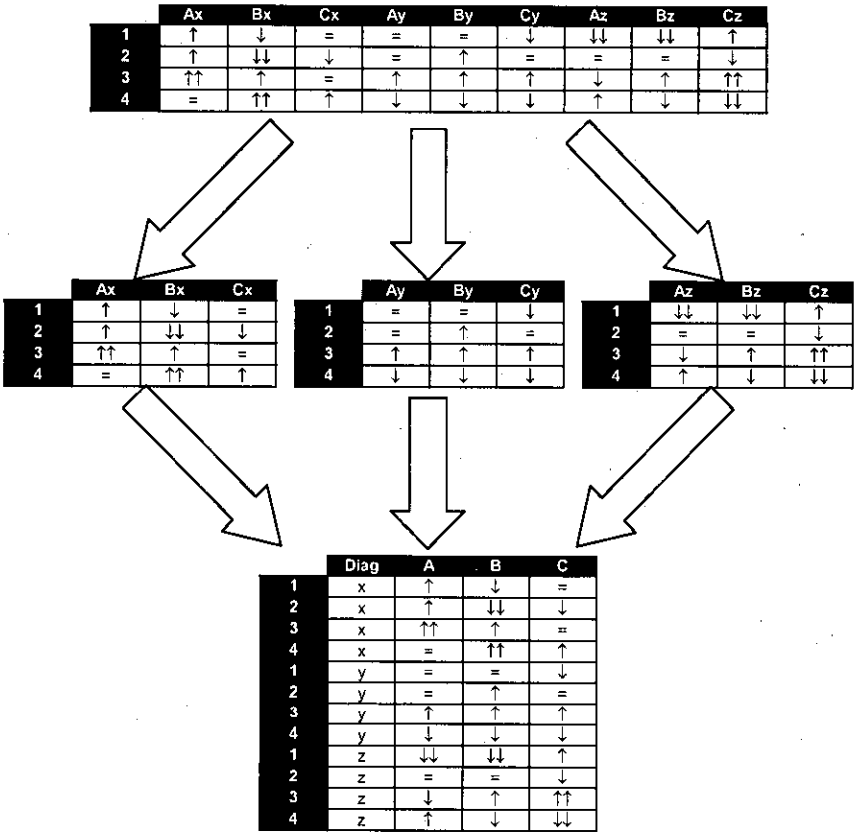


Figura 8.12. Conversión de una base de datos ECDC en varias ECDF y posteriormente unión de estas en una única base de datos ECDF

Para el caso EFDF podemos escoger aquellas filas que pertenecen a un único diagnóstico y unirlas en una misma base de datos. Esto permite que se formen tantas bases de datos EFDC como diagnósticos haya. Después es fácil unir estas bases de datos en una única base de datos EFDC (Figura 8.13).

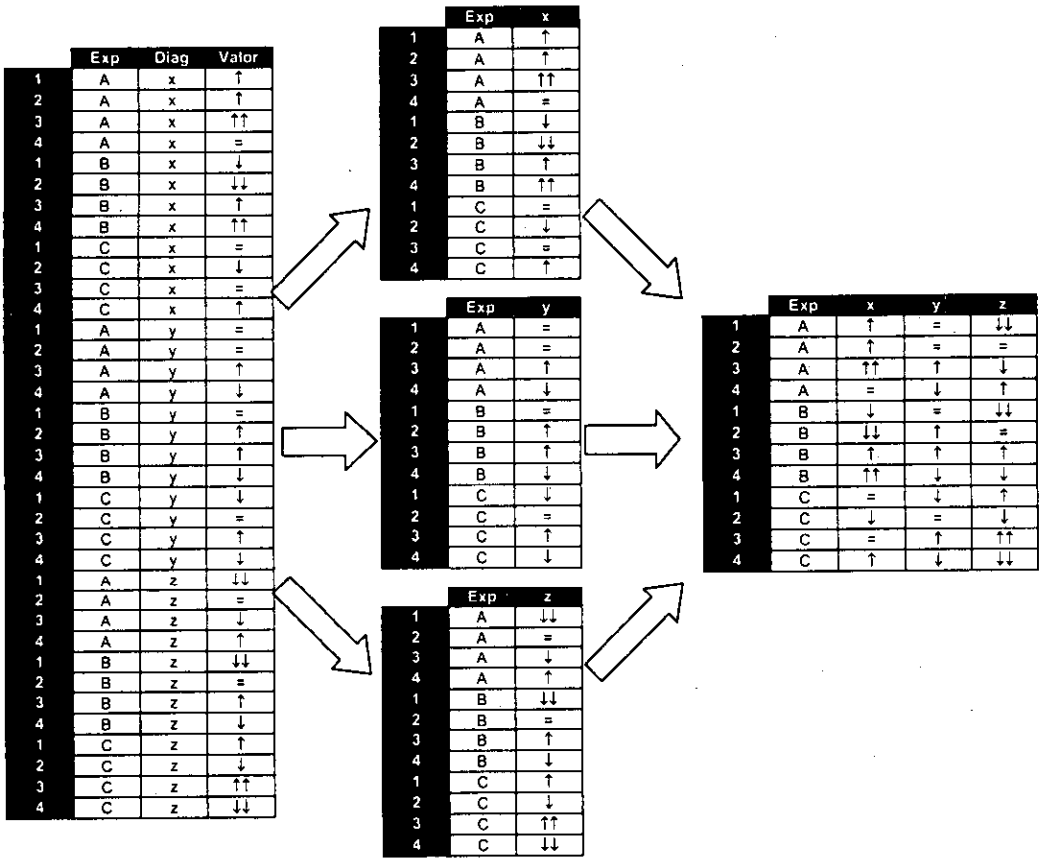


Figura 8.13. Conversión de una base de datos EFDF en varias EFDC y posterior unión en una única EFDC.

Ya que las bases de datos ECDC y EFDF son poco comunes y pueden ser fácilmente convertibles a bases de datos ECDF o EFDC, la primera versión de SHIVA sólo soporta estos últimos modelos. De todas formas está pensado que versiones posteriores puedan trabajar con los cuatro modelos de bases de datos.

Los datos de la Tabla 6.2 en la que se relacionaban los diagnósticos de cuatro expertos humanos y un sistema experto constituyen un claro ejemplo de una tabla ECDF. En este caso sería necesario añadir una nueva columna en cuyas filas se indicara el nombre del diagnóstico a tratar.

El primer paso del asistente para esta base de datos (almacenada en el fichero DATOS.DBF) puede verse en la Figura 8.14.

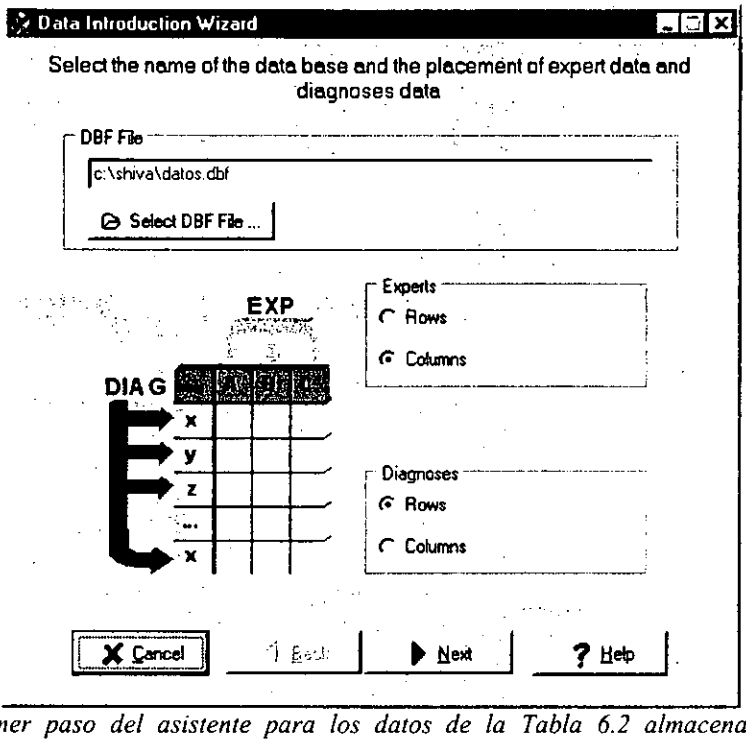


Figura 8.14. Primer paso del asistente para los datos de la Tabla 6.2 almacenada en el fichero DATOS.DBF

8.4.1.2. Tipado del los campos de la base de datos

El segundo paso del asistente consiste en asignar un determinado tipo a los campos que forman la base de datos de validación. Existen siete posibles tipos que pueden tomar los campos, y que se extraen de las cuatro posibles disposiciones de la base de datos (ver Figura 8.15):

1) *Experto*: El nombre del campo indica el nombre del experto mientras que sus valores indican los resultados del experto indicado. Se utiliza en ficheros ECDF.

2) *Campo de diagnósticos*: El nombre del campo no indica nada mientras que sus valores indicarán el diagnóstico que se está tratando en cada caso. Se utiliza en ficheros ECDF y EFDF.

3) *Diagnóstico*: El nombre del campo indica el nombre del diagnóstico mientras que sus valores indican los resultados sobre el diagnóstico indicado. Se utiliza en ficheros EFDC.

4) *Campo de expertos*: El nombre del campo no indica nada mientras que sus valores indicarán el experto que está tratando cada caso. Se utiliza en ficheros EFDC y EFDF.

5) *Experto y diagnóstico*: El nombre del campo indica el nombre del experto y del diagnóstico, también es posible especificar este nombre desde la herramienta. Los valores indicarán el resultado del experto para el diagnóstico indicado. Se utiliza en ficheros ECDC.

6) *Valores*: El nombre del campo no indica nada, mientras que sus valores indican resultados realizados por expertos sobre unos determinados diagnósticos. Se utiliza en ficheros EFDF.

7) *Otros*: Otro tipo de campos de la base de datos que no intervienen en la validación.

Otros	Campo diagnósticos	Experto	Experto	Experto
Num	Diag	A	B	C
1	x	↑	↓	=
2	x	↑	↓↓	↓
3	x	↑↑	↑	=
4	x	=	↑↑	↑
1	y	=	=	↓
2	y	=	↑	=
3	y	↑	↑	↑
4	y	↓	↓	↓
1	z	↓↓	↓↓	↑
2	z	=	=	↓
3	z	↓	↑	↑↑
4	z	↑	↓	↓↓

ECDF

Otros	Campo expertos	Diagnóstico	Diagnóstico	Diagnóstico
Num	Exp	x	y	z
1	A	↑	=	↓↓
2	A	↑	=	=
3	A	↑↑	↑	↓
4	A	=	↓	↑
1	B	↓	=	↓↓
2	B	↓↓	↑	=
3	B	↑	↑	↑
4	B	↑↑	↓	↓
1	C	=	↓	↑
2	C	↓	=	↓
3	C	=	↑	↑↑
4	C	↑	↓	↓↓

EFDC

Otros	Campo expertos	Campo diagnósticos	Valores
Num	Exp	Diag	Valor
1	A	x	↑
2	A	x	↑
3	A	x	↑↑
4	A	x	=
1	B	x	↓
2	B	x	↓↓
3	B	x	↑
4	B	x	↑↑
1	C	x	=
2	C	x	↓
3	C	x	=
4	C	x	↑
1	A	y	=
2	A	y	=
3	A	y	↑
4	A	y	↓
1	B	y	=
2	B	y	↑
3	B	y	↑
4	B	y	↓
1	C	y	↓
2	C	y	=
3	C	y	↑
4	C	y	↓
1	A	z	↓↓
2	A	z	=
3	A	z	↓
4	A	z	↑
1	B	z	↓↓
2	B	z	=
3	B	z	↑
4	B	z	↓
1	C	z	↑
2	C	z	↓
3	C	z	↑↑
4	C	z	↓↓

EFDF

ECDC

Otros	Experto diag.	Experto diag.	Experto diag.	Experto diag.	Experto diag.	Experto diag.	Experto diag.	Experto diag.	Experto diag.
Num	Ax	Bx	Cx	Ay	By	Cy	Az	Bz	Cz
1	↑	↓	=	=	=	↓	↓↓	↓↓	↑
2	↑	↓↓	↓	=	↑	=	=	=	↓
3	↑↑	↑	=	↑	↑	↑	↓	↑	↑↑
4	=	↑↑	↑	↓	↓	↓	↑	↓	↓↓

Figura 8.15. Tipos de los campos de los distintos modelos de bases de datos.

La ventana del asistente permitirá solamente acceder a los tipos que correspondan con la modalidad de la base de datos que se ha indicado en el paso anterior. Por ejemplo para la validación del fichero DATOS.DBF, veíamos que se trataba de un fichero ECDF por lo que la ventana del asistente tendra la siguiente configuración (Figura 8.16):

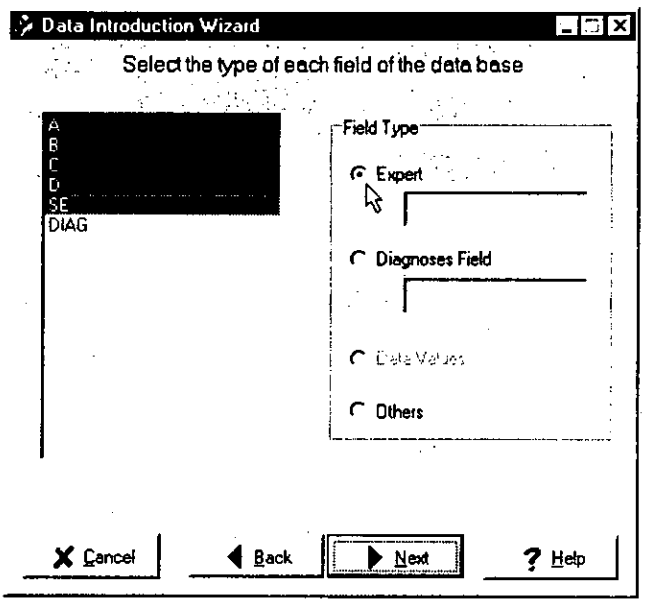


Figura 8.16. Segunda ventana del asistente en la que se introducen los tipos de los campos de la base de datos.

8.4.1.3. Establecimiento de los pesos y del orden entre las categorías

El siguiente paso es establecer la relación que existe entre las categorías en las que se divide un diagnóstico. En primer lugar es necesario incluir aquellas categorías que puedan no aparecer en la base de datos de validación.

Una vez que hemos completado el cuadro de categorías es necesario indicar su orden (ejemplo de DATOS.DBF en la Figura 8.17). Esto sólo tendrá sentido en caso de categorías ordinales. Si las categorías son nominales el orden que impongamos no importará porque las medidas que utilizaremos no tendrán en cuenta el orden.

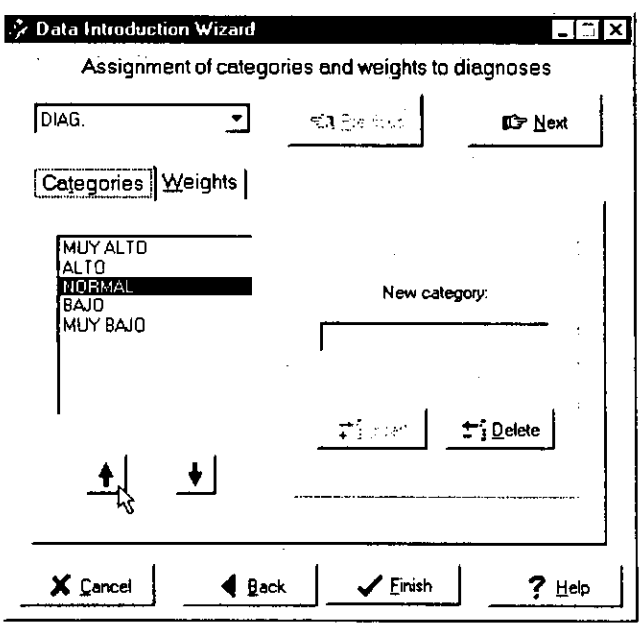


Figura 8.17. Tercera ventana del asistente en donde incluimos nuevas categorías y establecemos el orden de las mismas.

Algunas medidas, como kappa ponderada, establecen una ponderación entre los distintos tipos de discrepancias. Estos pesos pueden incluirse también en la tercera

ventana del asistente de introducción de datos (ejemplo de DATOS.DBF en la Figura 8.18). Existen dos configuraciones por defecto: que los pesos sigan una progresión aritmética, o que sigan una progresión geométrica. De todas formas puede ponerse cualquier combinación que se desee.

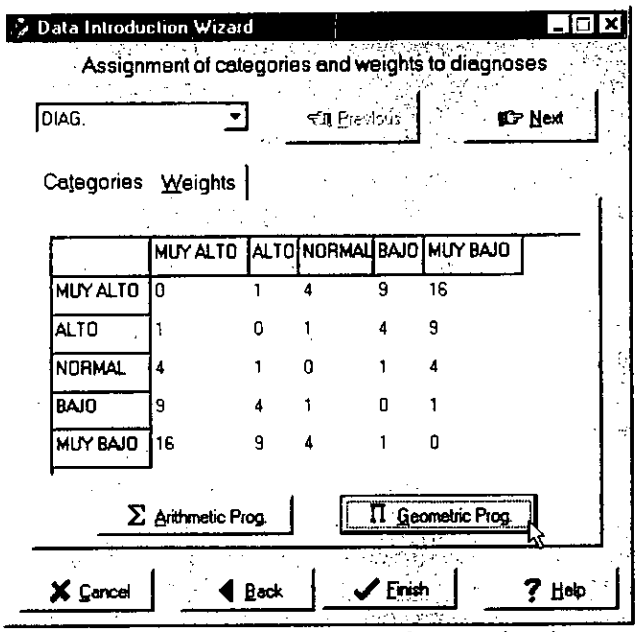


Figura 8.18. Tercera ventana del asistente en la que ponderamos las discrepancias que se producen entre las distintas categorías.

8.4.1.4. Ficheros VAL

El proceso de añadir información adicional a un fichero dBase es sencillo, pero sería bastante molesto tener que repetirlo cada vez que se quisiera abrir la base de datos de validación. Por ello SHIVA permite guardar esta información adicional en un fichero con extensión VAL.

El objetivo de los ficheros VAL no es almacenar los datos de la base de datos, sino almacenar aquella información necesaria para realizar la validación. El fichero VAL mantiene una referencia al fichero DBF del cual se ha extraído la información, por lo que ambos son necesarios para realizar la validación.

Los ficheros VAL son simples ficheros de texto con una estructura determinada. Esto permite que no sea necesario emplear la herramienta SHIVA para crearlos o modificarlos, simplemente basta con un sencillo editor de texto y con el conocimiento de la gramática en la que se basan. Como la información contenida en un fichero VAL no es mucha, su tamaño no suele ser elevado.

Gramática de los ficheros VAL

La gramática de los ficheros VAL se compone de 22 símbolos terminales que pueden aparecer en la entrada, 16 símbolos no terminales que agrupan a los símbolos terminales y se representan en mayúsculas, y de las siguientes 28 reglas gramaticales (en las que λ representa la cadena vacía):

```
(1)  S            →  CABSHIVA ARCHIVODBF EXP DIAG LISTACAMPOS
(2)  CABSHIVA     →  [ str_shiva ]
(3)  ARCHIVODBF   →  str_archivo_dbf = etiqueta
(4)  EXP          →  str_expertos = etiqueta
(5)  DIAG         →  str_diags = etiqueta
(6)  LISTACAMPOS  →  # numero = etiqueta : TIPO RESTOCAMPOS
(7)  RESTOCAMPOS  →  # numero = etiqueta : TIPO RESTOCAMPOS
(8)               |  λ
(9)  TIPO         →  str_otros
(10)              |  str_expertos
(11)              |  str_campo_expertos
(12)              |  str_diags DATOSVAL
(13)              |  str_campo_diags LISTADIAGS
(14)              |  str_exp_diag EXP DIAG LISTADIAGS
(15)              |  str_valores
(16) LISTADIAGS   →  etiqueta DATOSVAL RESTODIAGS
(17) RESTODIAGS   →  etiqueta DATOSVAL RESTODIAGS
(18)              |  λ
(19) DATOSVAL     →  LISTAVAL PESOS
(20) LISTAVAL     →  @ numero = etiqueta RESTOVAL
(21) RESTOVAL     →  @ numero = etiqueta RESTOVAL
(22)              |  λ
(23) PESOS        →  str_pesos = TIPOPESOS
(24) TIPOPESOS    →  prog_geo
(25)              |  prog_arit
(26)              |  numero RESTOPESOS
(27) RESTOPESOS   →  , numero RESTOPESOS
(28)              |  λ
```

En primer lugar vemos que la gramática busca la cadena [SHIVA] al principio del fichero. Esto le permite identificar al fichero como un fichero VAL. Seguidamente se especifica el nombre del fichero DBF y la localización en filas o en columnas de los expertos y los diagnósticos. Después sigue una enumeración de los campos de la base de datos con sus tipos. Dependiendo del tipo de cada campo la información asociada puede variar. Así, por ejemplo, si el campo es un diagnóstico habrá que incluir toda la información sobre las categorías de ese diagnóstico.

Como ejemplo de fichero VAL podemos ver el generado para la base de validación DATOS.DBF (Figura 8.19)

```
[SHIVA]

DBF File = 'datos.dbf'

Experts   = 'Columns'
Diagnosis = 'Rows'

#1 = 'A' : Expert
#2 = 'B' : Expert
#3 = 'C' : Expert
#4 = 'D' : Expert
#5 = 'SE' : Expert
#6 = 'DIAG' : DiagsField

  'DIAG.'

  @1 = 'MUY BAJO'
  @2 = 'BAJO'
  @3 = 'NORMAL'
  @4 = 'ALTO'
  @5 = 'MUY ALTO'

  Weights = Geometric
```

Figura 8.19. Fichero VAL para la base de validación DATOS.DBF

Interprete de ficheros VAL

Los ficheros VAL, al ser ficheros ASCII, necesitan de un pequeño intérprete para que sus datos sean accesibles por SHIVA. Este intérprete se compone de tres partes:

- *Analizador léxico*: Identifica en la cadena de entrada los *tokens* o componentes léxicos con significado. Por ejemplo si encuentra la cadena "DBF File" devolverá el token *str_archivo_dbf*. Se realiza mediante una función de reconocimiento de tokens.

- *Analizador sintáctico*: Comprueba que los tokens identificados por el analizador léxico siguen la gramática especificada. Se implementa mediante un análisis sintáctico para gramáticas LL(1)

- *Analizador semántico*: Verifica que la utilización de la gramática es la correcta. Se va ejecutando a medida que se van identificando partes concretas de la gramática.

El análisis sintáctico para gramáticas LL(1)

En la Figura 8.20 se muestra un modelo de analizador sintáctico no predictivo. En él podemos identificar un buffer de entrada, una pila, una tabla de análisis sintáctico, una cadena de salida y, por supuesto, el programa de análisis.

El funcionamiento del programa de análisis es el siguiente (Aho, Sethi y Ullman, 1990): compara el valor que tiene en la pila (X), y el valor que tiene en la cadena de entrada (a), y realiza las siguientes acciones:

1. Si $X = a = \$$, siendo \$ el carácter de terminación, el analizador sintáctico se detiene y anuncia el éxito de la realización del análisis.

2. Si $X = a \neq \$$, el analizador sintáctico saca a X de la pila y mueve el apuntador de la entrada al siguiente símbolo.

3. Si X es un no terminal, el programa consulta la entrada $M[X, a]$ de la tabla M de análisis sintáctico. Esta entrada será o una producción de X de la gramática, o una entrada de error. Si, por ejemplo, $M[X, a] = \{X \rightarrow UVW\}$, el analizador sintáctico substituye la X de la cima de la pila por WVU (con U en la cima). Si $M[X, a] = error$, el analizador sintáctico llama a una rutina de recuperación de error.

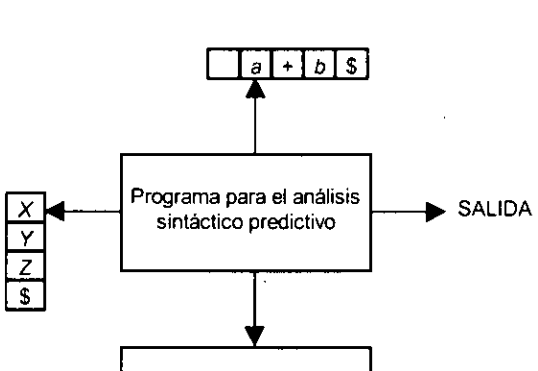


Figura 8.20. Analizador sintáctico descendente predictivo no recursivo.

La gramática que define a los ficheros VAL es del tipo LL(1). Se puede demostrar que, a partir de gramáticas LL(1), puede construirse una tabla que permita utilizar el analizador sintáctico de la Figura 8.20.

La construcción de la tabla de análisis se basa en la identificación de los conjuntos PRIMERO y SIGUIENTE para los elementos no terminales de la gramática. El conjunto PRIMERO de un elemento X identifica a todos los terminales que comienzan las cadenas derivadas de X, mientras que el conjunto SIGUIENTE de un no terminal X identifica al conjunto de terminales que pueden aparecer inmediatamente a la derecha de X en alguna frase. Los conjuntos PRIMERO y SIGUIENTE para la gramática de SHIVA son:

	PRIMERO	SIGUIENTE
S	[	$
CABSHIVA	[	str_archivo_dbf
ARCHIVODBF	str_archivo_dbf	str_expertos
EXP	str_expertos	str_diags
DIAG	str_diags	# etiqueta
LISTACAMPOS	#	$
RESTOCAMPOS	# λ	$
TIPO	str_otros str_expertos str_campo_expertos str_diags str_campo_diags str_exp_diag str_valores	# $
LISTADIAGS	etiqueta	# $
RESTODIAGS	etiqueta λ	# $
DATOSVAL	@	# $ etiqueta
LISTAVAL	@	str_pesos
RESTOVAL	@ λ	str_pesos
PESOS	str_pesos	# $ etiqueta
TIPOPESOS	prog_geo prog_arit numero	# $ etiqueta
RESTOPESOS	, λ	# $ etiqueta

Tabla 8.1. Conjuntos PRIMERO y SIGUIENTE para los no terminales de la gramática de SHIVA.

En base a estos dos conjuntos es sencillo construir la tabla de análisis M para realizar el análisis sintáctico de la Figura 8.20 y que se muestra en la Tabla 8.2. Para

más detalles sobre como construir los conjuntos PRIMERO y SIGUIENTE, o sobre como derivar la tabla *M* puede consultarse (Aho, Sethi y Ullman, 1990).

M	S	CAB SHIVA	ARCHIVO DBF	EXP	DIAG	LISTA CAMPOS	RESTO CAMPOS	TIPO	LISTA DIAGS	RESTO DIAGS	DATOS VAL	LISTA VAL	RESTO VAL	PESOS	TIPO PESOS	RESTO PESOS
[	1	2														
str_ shiva																
]																
str_ archivo_ dbf			3													
=																
etiqueta									16	17						28
#						6	7			18						28
numero															26	
:																
str_ otros								9								
str_ expertos				4				10								
str_ campo_ expertos								11								
str_ diags					5			12								
str_ campo_ diags								13								
str_exp_ diag								14								
str_ valores								15								
@											19	20	21			
str_ pesos													22	23		
prog_ geo															24	
prog_ arit															25	
.																27
$							8			18						28

Tabla 8.2. Matriz M de la gramática de SHIVA para el desarrollo de un análisis sintáctico descendente predictivo no recursivo. Los números indican las producciones de la gramática y las casillas en blanco errores sintácticos.

El análisis léxico del fichero VAL de la Figura 8.19 nos permite obtener la siguiente cadena de tokens:

[str_shiva] str_archivo_dbf = etiqueta str_expertos = etiqueta str_diags = etiqueta # numero = etiqueta : str_expertos # numero = etiqueta : str_expertos # numero = etiqueta : str_expertos # numero = etiqueta : str_expertos # numero = etiqueta : str_campo_diags etiqueta @ numero = etiqueta @ numero = etiqueta @ numero = etiqueta @ numero = etiqueta str_pesos = prog_geo $

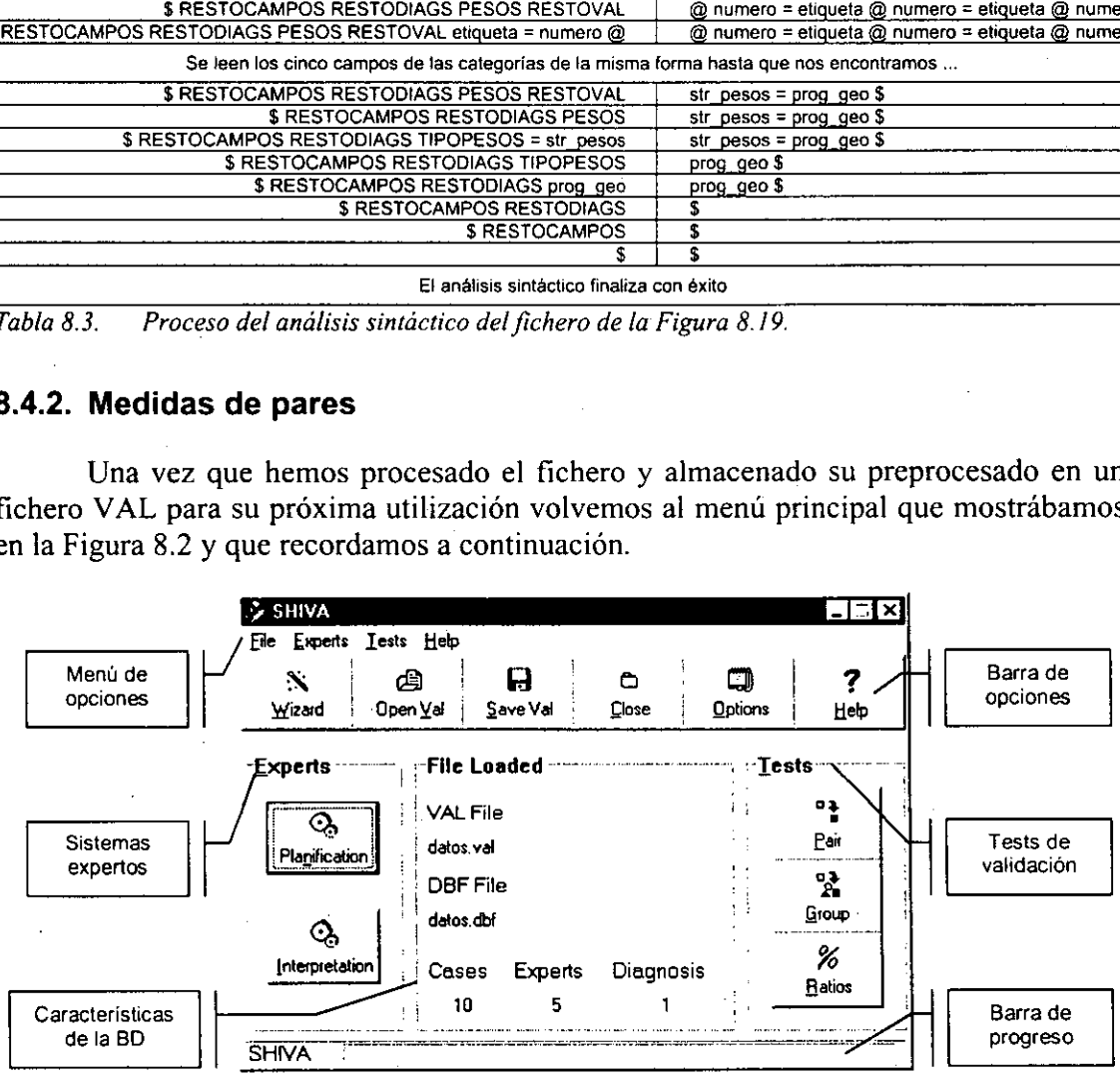
El proceso de aplicar el analizador sintáctico puede verse en la Tabla 8.3:

CONTENIDO DE LA PILA (primer elemento ↓)	↓ primer elemento (CONTENIDO DE LA ENTRADA)
$ S	[str_shiva] str_archivo_dbf = etiqueta str_expertos = ...
$ LISTACAMPOS DIAG EXP ARCHIVODBF CABSHIVA	[str_shiva] str_archivo_dbf = etiqueta str_expertos = ...
$ LISTACAMPOS DIAG EXP ARCHIVODBF] str_shiva [	[str_shiva] str_archivo_dbf = etiqueta str_expertos = ...
$ LISTACAMPOS DIAG EXP ARCHIVODBF	str_archivo_dbf = etiqueta str_expertos = etiqueta ...
$ LISTACAMPOS DIAG EXP etiqueta = str_archivo_dbf	str_archivo_dbf = etiqueta str_expertos = etiqueta ...
$ LISTACAMPOS DIAG EXP	str_expertos = etiqueta str_diags = etiqueta # numero = ...
$ LISTACAMPOS DIAG etiqueta = str_expertos	str_expertos = etiqueta str_diags = etiqueta # numero = ...
$ LISTACAMPOS DIAG	str_diags = etiqueta # numero = etiqueta : str_expertos ...
$ LISTACAMPOS etiqueta = str_diags	str_diags = etiqueta # numero = etiqueta : str_expertos ...
$ LISTACAMPOS	# numero = etiqueta : str_expertos # numero = etiqueta : ...
$ RESTOCAMPOS TIPO : etiqueta = numero #	# numero = etiqueta : str_expertos # numero = etiqueta : ...
$ RESTOCAMPOS TIPO	str_expertos # numero = etiqueta : str_expertos # ...
$ RESTOCAMPOS str_expertos	str_expertos # numero = etiqueta : str_expertos # ...
Se leen los cinco campos de los expertos de la misma forma hasta que nos encontramos ...	
$ RESTOCAMPOS	# numero = etiqueta : str_campo_diags etiqueta @ ...
$ RESTOCAMPOS TIPO : etiqueta = numero #	# numero = etiqueta : str_campo_diags etiqueta @ ...
$ RESTOCAMPOS TIPO	str_campo_diags etiqueta @ numero = etiqueta @ ...
$ RESTOCAMPOS LISTADIAGS str_campo_diags	str_campo_diags etiqueta @ numero = etiqueta @ ...
$ RESTOCAMPOS LISTADIAGS	etiqueta @ numero = etiqueta @ numero = etiqueta ...
$ RESTOCAMPOS RESTODIAGS DATOSVAL etiqueta	etiqueta @ numero = etiqueta @ numero = etiqueta ...
$ RESTOCAMPOS RESTODIAGS DATOSVAL	@ numero = etiqueta @ numero = etiqueta @ numero ...
$ RESTOCAMPOS RESTODIAGS PESOS LISTAVAL	@ numero = etiqueta @ numero = etiqueta @ numero ...
$ RESTOCAMPOS RESTODIAGS PESOS RESTOVAL etiqueta = numero @	@ numero = etiqueta @ numero = etiqueta @ numero ...
$ RESTOCAMPOS RESTODIAGS PESOS RESTOVAL	@ numero = etiqueta @ numero = etiqueta @ numero ...
$ RESTOCAMPOS RESTODIAGS PESOS RESTOVAL etiqueta = numero @	@ numero = etiqueta @ numero = etiqueta @ numero ...
Se leen los cinco campos de las categorías de la misma forma hasta que nos encontramos ...	
$ RESTOCAMPOS RESTODIAGS PESOS RESTOVAL	str_pesos = prog_geo $
$ RESTOCAMPOS RESTODIAGS PESOS	str_pesos = prog_geo $
$ RESTOCAMPOS RESTODIAGS TIPOPESOS = str_pesos	str_pesos = prog_geo $
$ RESTOCAMPOS RESTODIAGS TIPOPESOS	prog_geo $
$ RESTOCAMPOS RESTODIAGS prog_geo	prog_geo $
$ RESTOCAMPOS RESTODIAGS	$
$ RESTOCAMPOS	$
$	$
El análisis sintáctico finaliza con éxito	

Tabla 8.3. Proceso del análisis sintáctico del fichero de la Figura 8.19.

8.4.2. Medidas de pares

Una vez que hemos procesado el fichero y almacenado su preprocesado en un fichero VAL para su próxima utilización volvemos al menú principal que mostrábamos en la Figura 8.2 y que recordamos a continuación.



Como vemos la parte central de la ventana resume las características de la base de datos abierta, es decir, el nombre de los ficheros VAL y DBF que almacenan los datos, el número de casos considerados, el número de expertos, y el número de diagnósticos.

En la parte derecha de la ventana se encuentran los botones que permiten realizar los tests de pares, los tests de grupo y los ratios de acuerdo. Pulsando sobre el botón de las medidas de pares nos aparece la ventana que mostramos en la Figura 8.21.

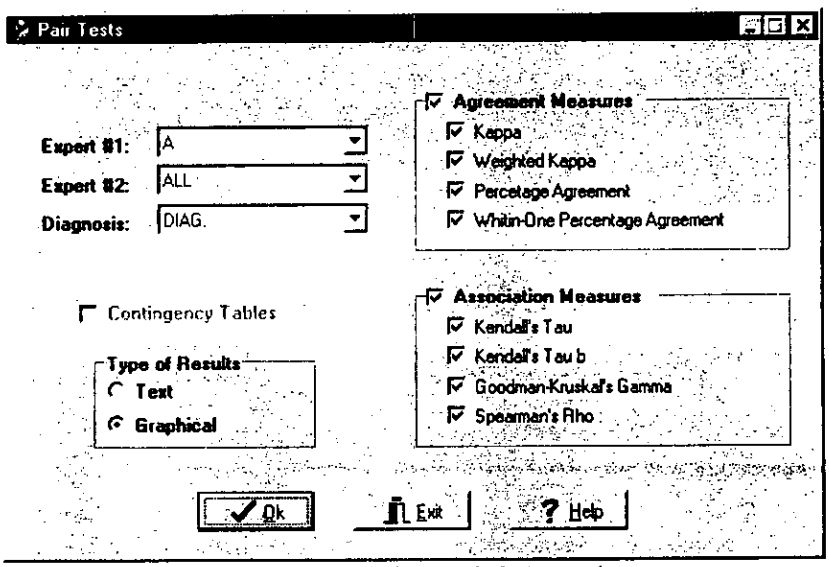


Figura 8.21 Ventana de selección de las características de los tests de pares.

Esta ventana nos permite escoger las opciones a aplicar a la hora de calcular los tests de pares. En primer lugar es necesario escoger los elementos de comparación, es decir, qué pares de expertos vamos a comparar y sobre qué diagnóstico. En este caso vemos que el primer experto seleccionado es el experto A, mientras que en el segundo hemos seleccionado la categoría "ALL" que indica que los resultado de A deben compararse con los del resto de expertos (sin incluir A evidentemente). Como diagnóstico seleccionamos el único que aparece en esta base de datos.

A la derecha de la ventana se incluyen los test de pares que se desean aplicar a los expertos y diagnóstico seleccionados. Los tests de pares se dividen en dos categorías: (1) medidas de acuerdo, que incluyen a kappa, kappa ponderada, el porcentaje de acuerdo y el porcentaje de acuerdo dentro de uno; y (2) medidas de asociación, que incluyen a tau y tau b de Kendall, gamma de Goodman-Kruskal y Rho de Spearman.

Por último, sólo nos queda escoger el formato de los resultado, es decir, si queremos que sean gráficas o documentos de texto en forma de tablas (en estos últimos se incluye también la opción de mostrar solamente los resultados numéricos o incluir también las tablas de contingencia).

El navegador de resultados gráficos se muestran en la ventana de la Figura 8.22 y su menú de opciones en la Figura 8.23. El navegador de los resultados de texto se muestran en la ventana de la Figura 8.24.

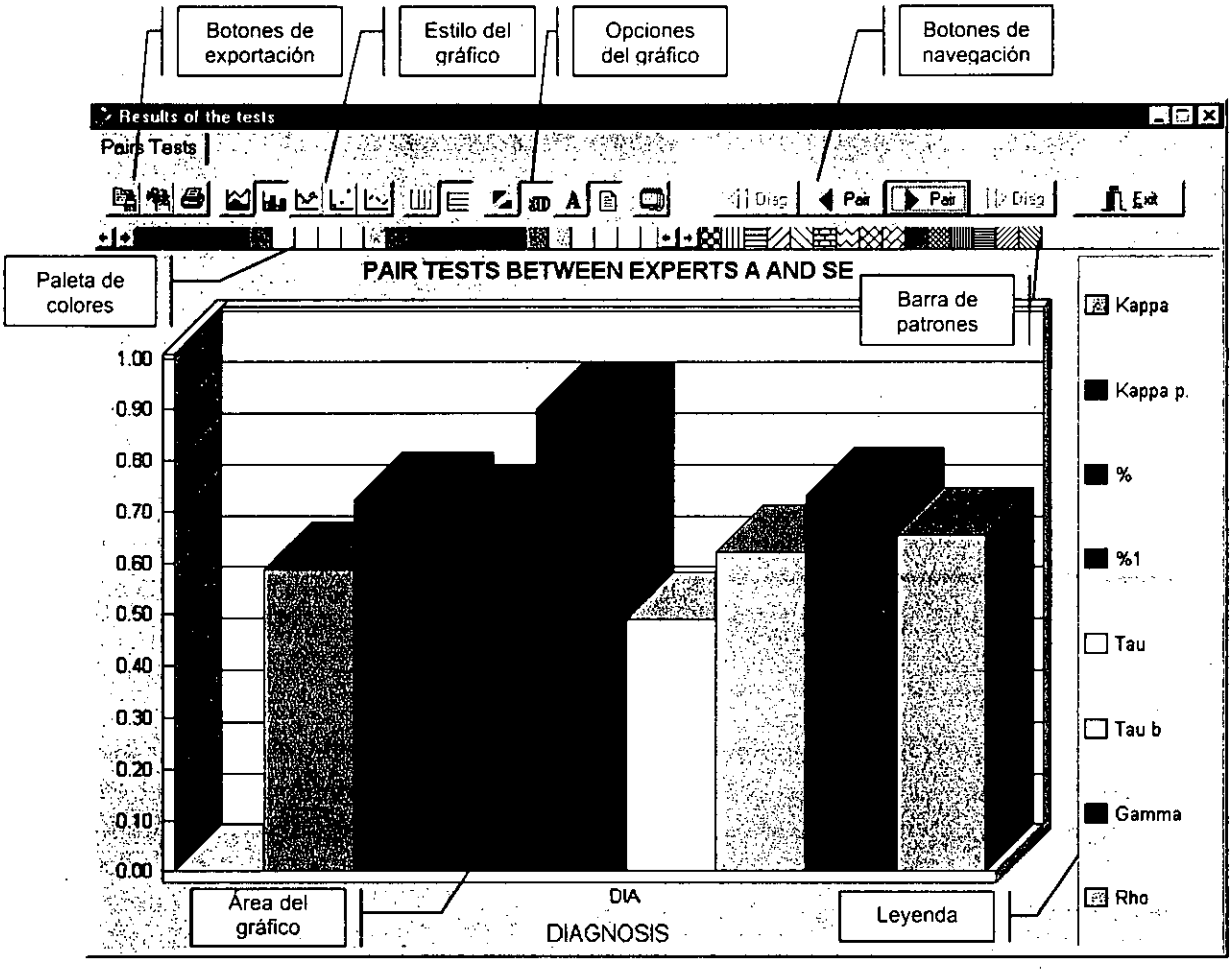


Figura 8.22 Resultados de los test de pares para los expertos A y SE en formato gráfico.

El navegador de resultados gráficos es utilizado por todos los tests para mostrar los resultados de una forma sencilla y gráfica que permita su rápida interpretación. El navegador se divide en una serie de áreas que pasaremos a explicar a continuación.

En primer lugar tenemos la *barra de herramientas*, que contiene los botones que nos permiten exportar el gráfico o modificar sus características. Esta barra se divide, a su vez, en varias áreas agrupando a los botones de funciones similares:

- *Botones de exportación*: utilizados para extraer el gráfico de la herramienta SHIVA. Existen tres posibilidades: guardar el gráfico en un bitmap, copiar el gráfico al portapapeles de Windows o imprimir el gráfico en una impresora.

- *Estilo del gráfico*: permiten modificar el estilo de las distintas series del gráfico. Las posibles opciones son: áreas, barras, líneas, puntos o curvas.

- *Opciones del gráfico*: permiten variar diversas opciones del gráfico como las líneas de división, la visión en colores o en patrones, la visión 2D o 3D, las fuentes y la leyenda. También se incluye un botón de

opciones que abre una pequeña ventana (Figura 8.23) que permite cambiar otras opciones del gráfico como la escala, la frecuencia de las líneas de división, el formato de los puntos, etc.

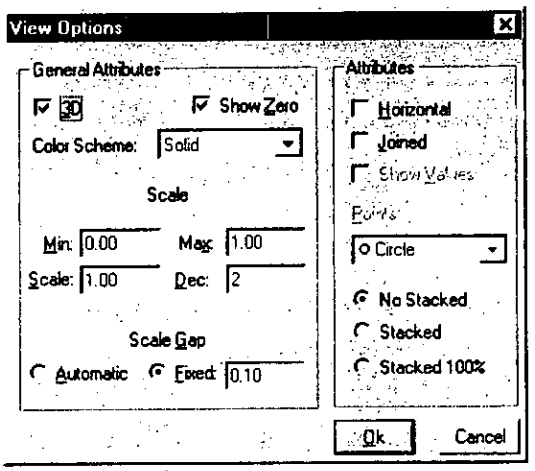


Figura 8.23 Ventana de opciones del gráfico.

- *Botones de navegación*: permiten navegar entre los distintos gráficos pertenecientes a los resultados de los tests de pares. Son de dos tipos: por un lado los botones de pares Test nos permiten ir variando entre los distintos pares de expertos que hemos seleccionado para realizar el test (por ejemplo, A-B, A-C, A-D, etc.). Los botones de diagnóstico Diag permiten variar entre los distintos diagnósticos que forman la base de datos. Como en este ejemplo sólo hemos incluido un diagnóstico los botones aparecen desactivados.

- *Botón de exit*: permite salir del navegador de resultados gráficos.

Debajo de la barra de herramientas se encuentra la *paleta de colores* y la *barra de patrones*. La paleta de colores permite cambiar los colores de las series del gráfico y la barra de patrones permite cambiar la trama de las series.

El resto de elementos del navegador son la leyenda del gráfico (que puede o no estar visible) y la propia área del gráfico.

El navegador de resultados en modo texto (Figura 8.24) utiliza la misma ventana que habíamos visto para mostrar los resultados del sistema experto de planificación. Es una ventana más sencilla que la del navegador de resultados gráficos y se compone simplemente de un área de texto (en donde se incluyen los resultados) y una barra de herramientas donde se incluyen botones para imprimir los resultados, guardarlos en disco, copiarlos al portapapeles, seleccionar todo el texto de área, o salir del navegador.

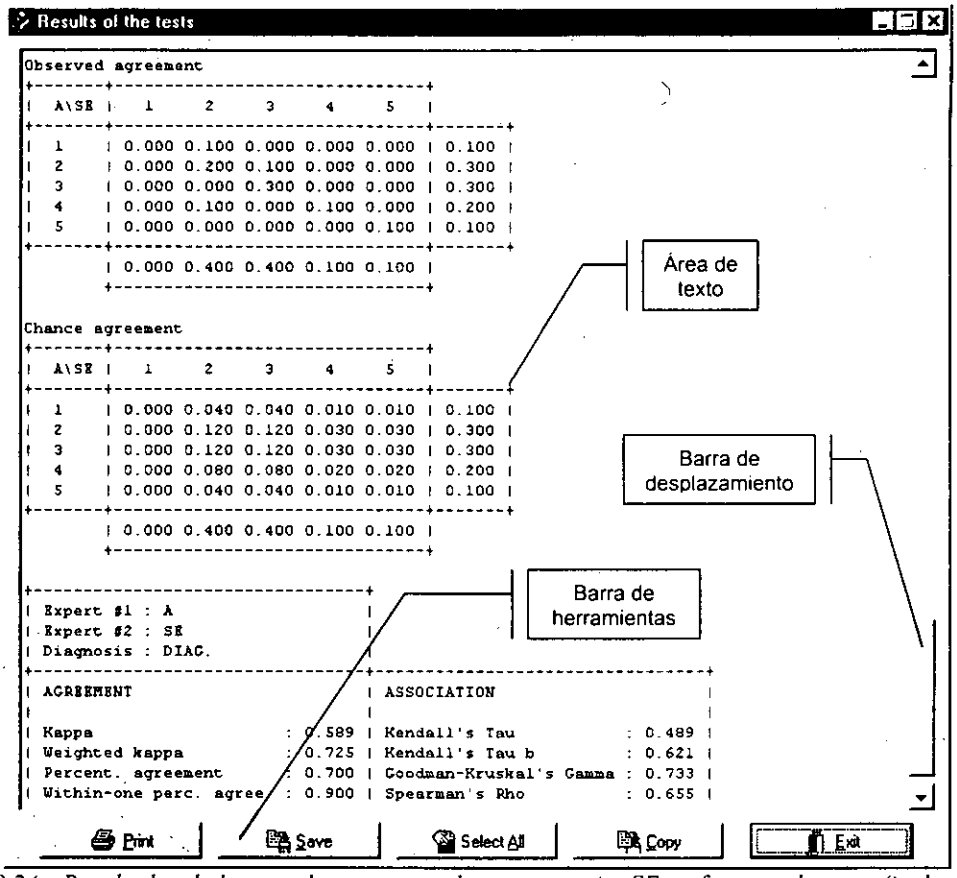


Figura 8.24 Resultados de los test de pares para los expertos A y SE en formato de texto (incluyendo las tablas de contingencia).

En este caso no existen botones de navegación porque todos los resultados seleccionados se muestran a la vez sobre el área de texto y pueden ser accedidos a través de la barra de desplazamiento.

8.4.3. Medidas de grupo

Para aplicar las medidas de grupo a la base de datos cargada es necesario acceder a la ventana de selección de las características de las medidas de grupo (Figura 8.25) a través del botón correspondiente en la ventana principal.

La ventana de las medidas de grupo se compone de:

- Una lista en la que seleccionar el diagnóstico para el cual se van a aplicar las medidas de grupo.
- Los tests de grupo que se van a aplicar a ese diagnóstico (incluyendo matrices resumen, análisis cluster, medidas de Williams, escalamiento multidimensional, y medidas de dispersión y tendencia). Las matrices resumen no son un test propiamente dicho, sólo permiten mostrar los resultados de los tests de pares de todos los expertos de forma conjunta en un único gráfico.
- Los test de pares en los que se han basado los tests de grupo.
- El tipo de resultados (gráficos o de texto).

- Una barra de herramientas con los botones de aceptar, salir y ayuda.

Existe también una opción que permite unir, en las matrices de resumen y en los test de Williams, los resultados de los distintos tests de pares de forma que se muestren en un solo gráfico como en series de datos distintas.

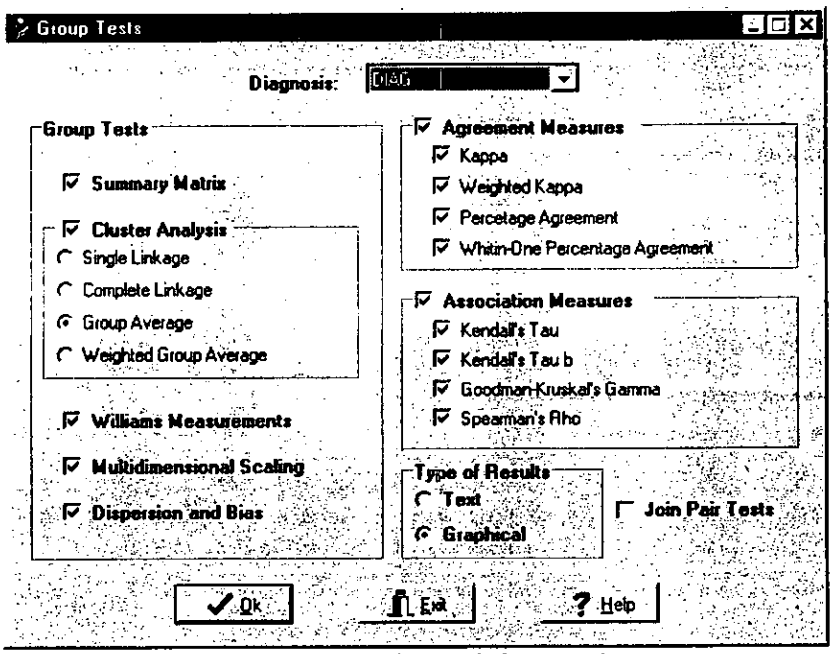


Figura 8.25 Ventana de selección de las características de los tests de grupo.

Los resultados de los tests de grupo se muestran en el mismo navegador de resultados que los tests de pares, pero incluyendo en el mismo varias páginas para poder mostrar los diferentes tests desarrollados.

Matrices resumen

En la Figura 8.26 podemos ver una ventana que contiene cinco hojas que corresponden a los cinco tests de grupo desarrollados. En este caso la hoja que está activa es la correspondiente a la matriz resumen, y muestra los resultados de todos los posibles pares de expertos para el test kappa ponderada. El único cambio con respecto a los test de pares es que, en los botones de navegación, los botones que cambiaban entre pares de expertos se han substituido por botones que nos permiten cambiar entre distintos tests. Si la opción "Join Pair Tests" esta activada, el gráfico muestra juntos los resultados de los distintos tests de pares, y los botones de test aparecen desactivados como se muestra en la Figura 8.27 para las medidas de acuerdo.

En los datos del ejemplo introducido se puede ver que los pares que presentan mejor acuerdo son A-B, A-SE y B-SE. También puede verse que los mayores valores se dan en el porcentaje de acuerdo dentro de uno, y los menores en kappa. Los valores de kappa ponderada y el porcentaje de acuerdo son muy similares, aunque dependerán de la distribución de pesos utilizada.

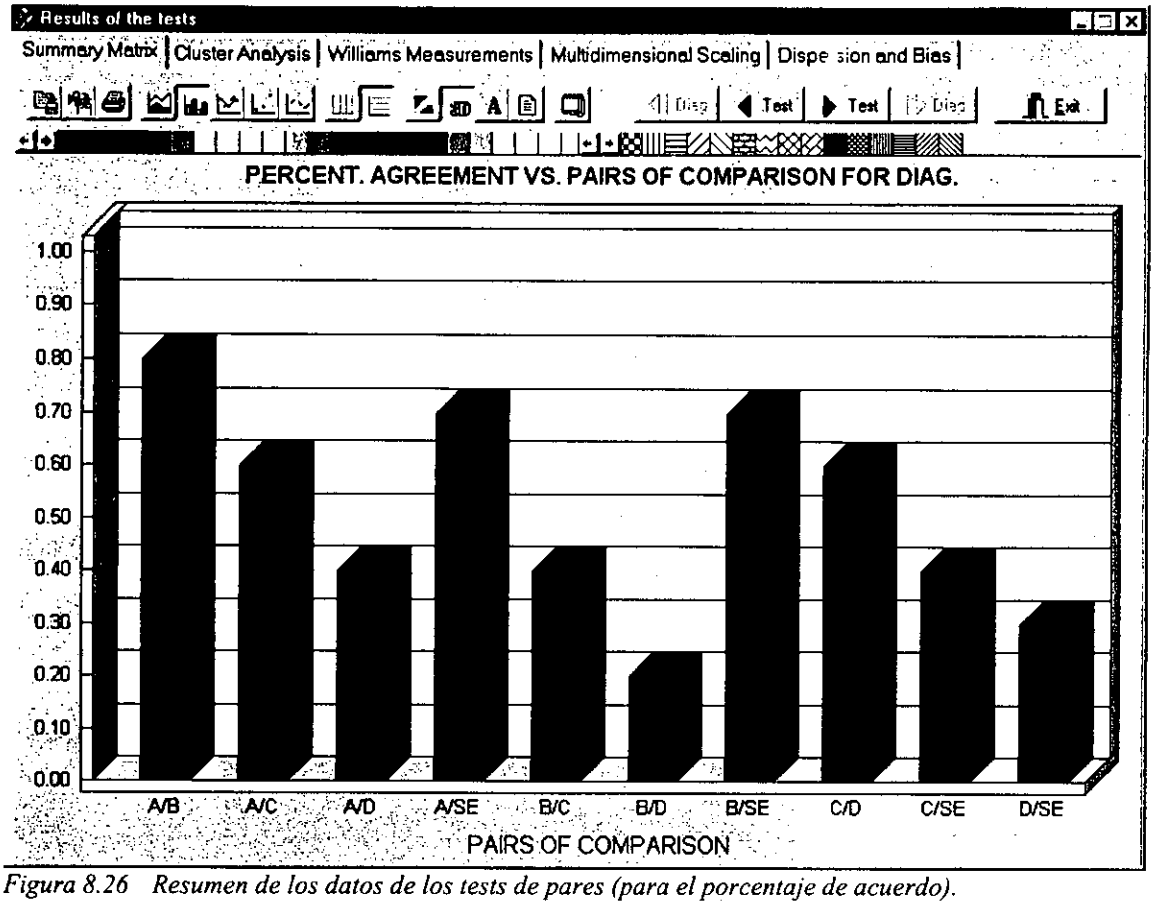


Figura 8.26 Resumen de los datos de los tests de pares (para el porcentaje de acuerdo).

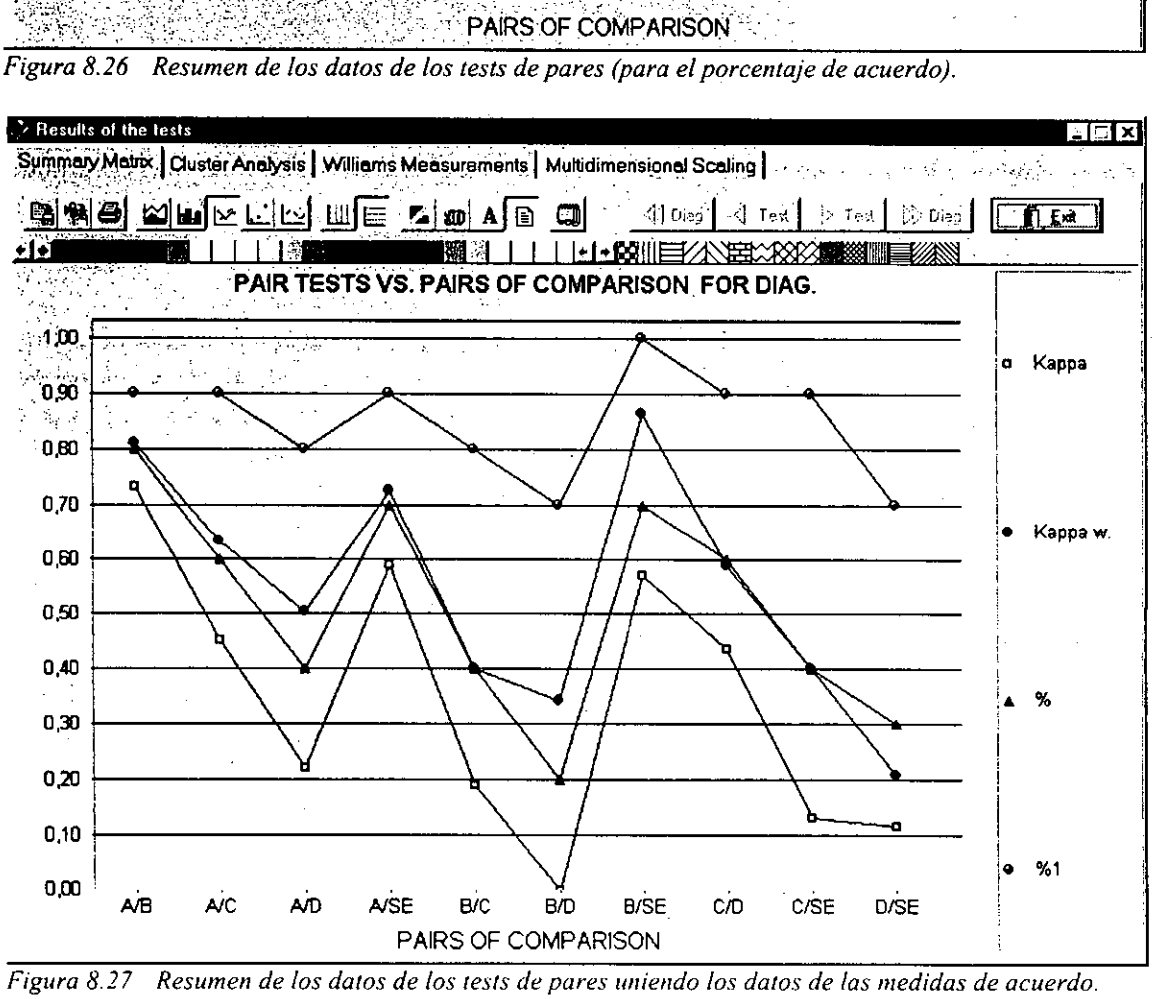


Figura 8.27 Resumen de los datos de los tests de pares uniendo los datos de las medidas de acuerdo.

Análisis cluster

En la Figura 8.28 podemos ver la hoja que representa los resultados del análisis cluster. En dicha hoja vuelven a aparecer los botones de exportación y los botones de navegación y, además, aparece un nuevo botón que permite aplicar el método para la detección del número ideal de clusters.

Dentro de lo que es el área del cluster distinguimos en primer lugar el título, después la regla que indica los niveles de unión del dendrograma, y el dendrograma en sí. Por último incluimos el nombre del algoritmo de clustering utilizado así como la correlación cofenética.

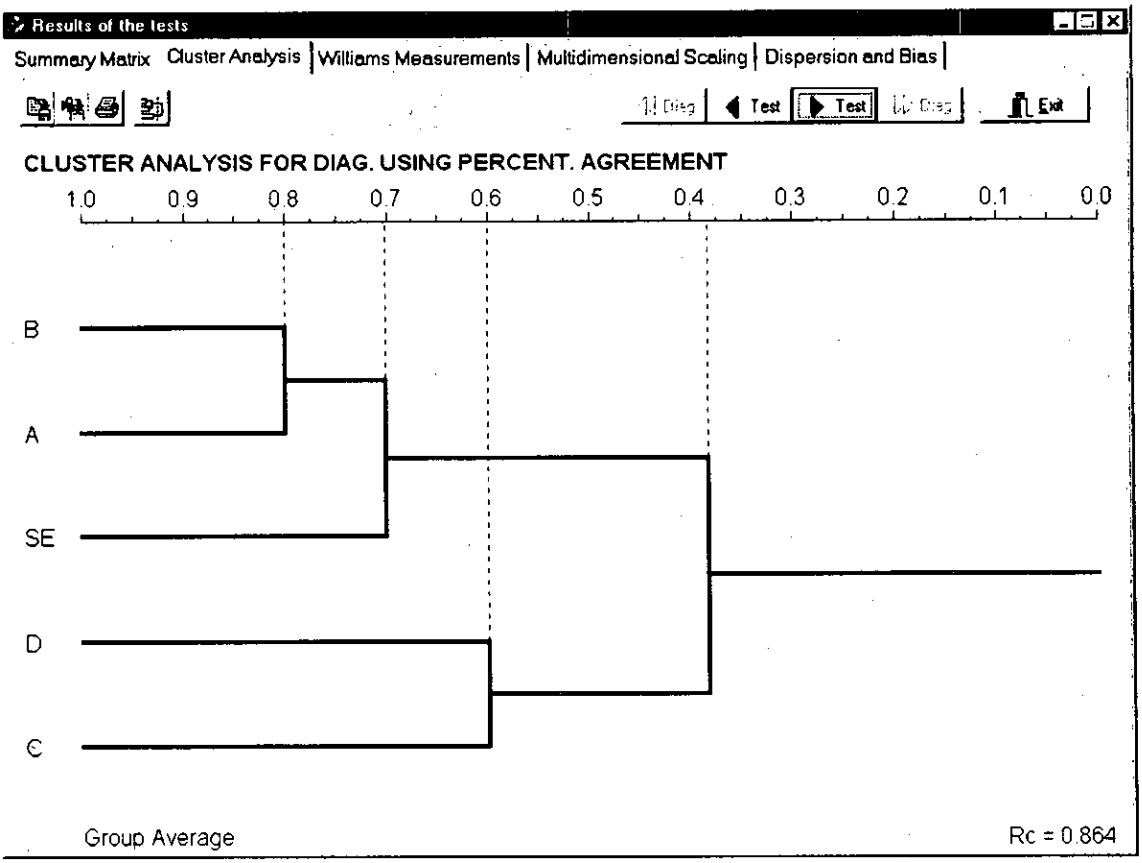


Figura 8.28 Resultados del análisis cluster para los datos del porcentaje de acuerdo.

Si deseamos saber qué partición de este dendrograma es la que responde de forma más fiel a los datos iniciales (en definitiva, determinar qué número de clusters sería el más adecuado) podemos utilizar el método ya descrito al definir los algoritmos de clustering (ver el apartado 6.2.2.8).

Este método consiste en analizar los niveles de fusión de los distintos clusters, seleccionar el paso en el que se produce la mayor diferencia entre los niveles de fusión y considerar que el número de clusters ideal era el existente antes de dar dicho paso.

Si pulsamos el botón de selección del número de clusters en la hoja del análisis cluster obtenemos el número de clusters ideal y una línea marcando donde se debería dividir el dendrograma, como se muestra en la Figura 8.29.

En este ejemplo podemos ver como se forman dos grupos, uno formado por los expertos A, B y SE; y otro formado por los experto C y D.

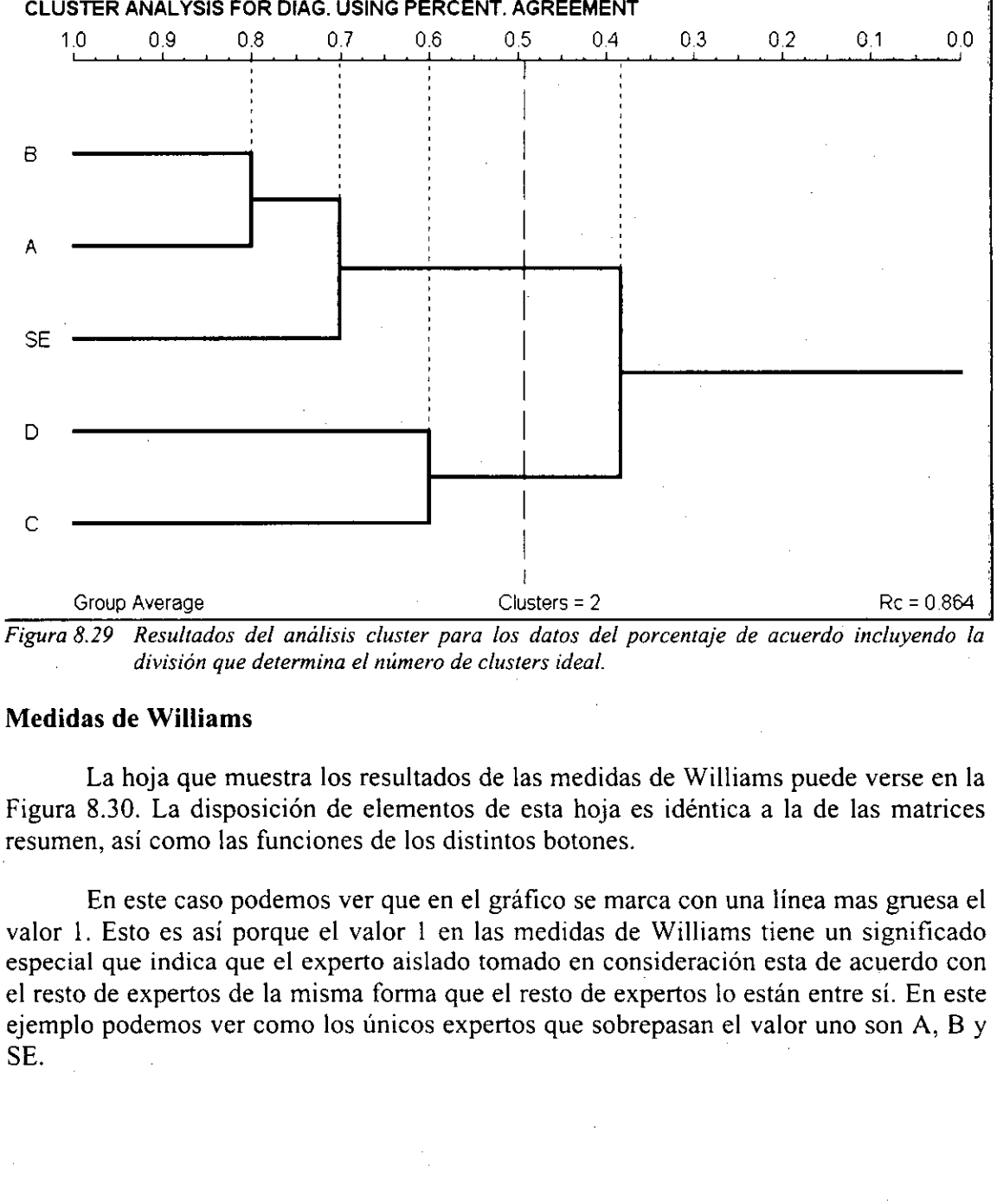


Figura 8.29 Resultados del análisis cluster para los datos del porcentaje de acuerdo incluyendo la división que determina el número de clusters ideal.

Medidas de Williams

La hoja que muestra los resultados de las medidas de Williams puede verse en la Figura 8.30. La disposición de elementos de esta hoja es idéntica a la de las matrices resumen, así como las funciones de los distintos botones.

En este caso podemos ver que en el gráfico se marca con una línea mas gruesa el valor 1. Esto es así porque el valor 1 en las medidas de Williams tiene un significado especial que indica que el experto aislado tomado en consideración esta de acuerdo con el resto de expertos de la misma forma que el resto de expertos lo están entre sí. En este ejemplo podemos ver como los únicos expertos que sobrepasan el valor uno son A, B y SE.

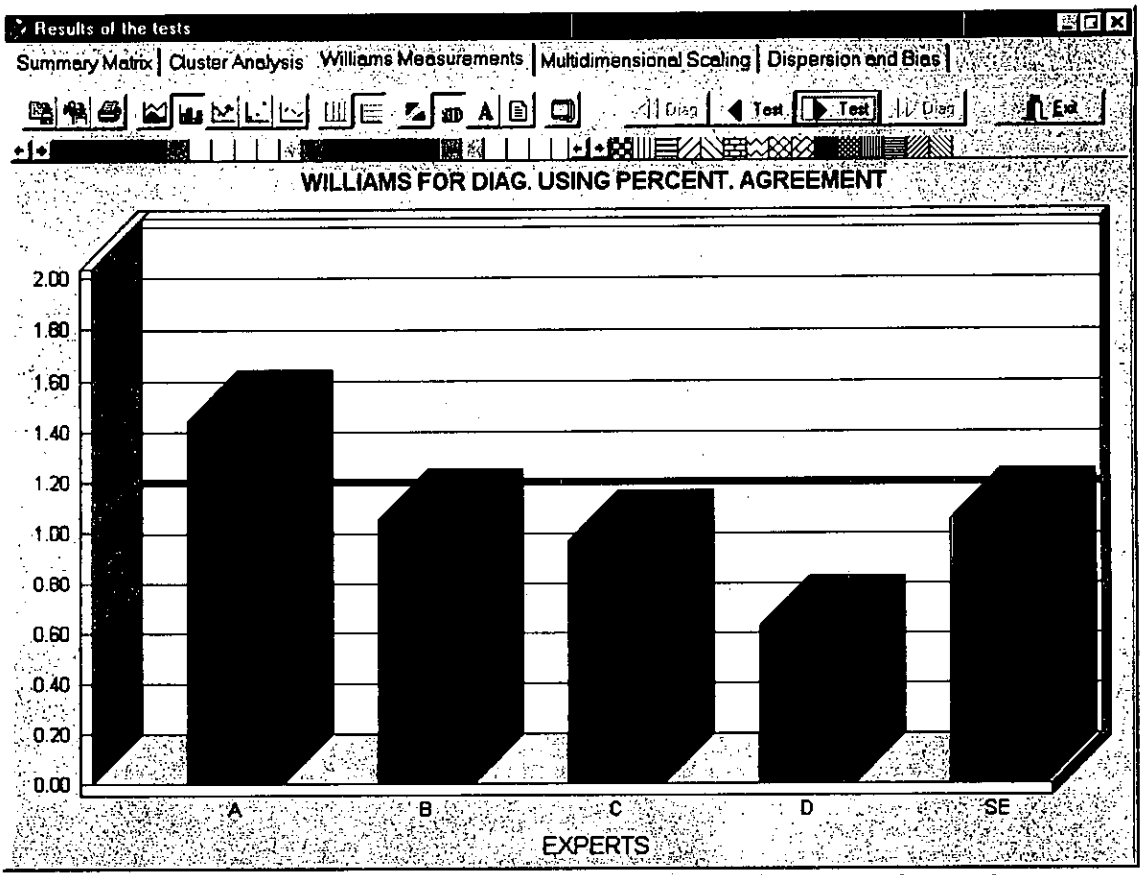


Figura 8.30 Resultados de las medidas de Williams para los datos del porcentaje de acuerdo.

Escalamiento multidimensional

La hoja que muestra los resultados del escalamiento multidimensional (MDS) puede verse en la Figura 8.31. En ella se vuelven a incluir los botones de exportación y navegación además de dos nuevos botones adicionales que comentaremos a continuación.

Como vemos, la gráfica representa los expertos como puntos en un espacio 2D. La disposición de los ejes es arbitraria, y el origen representa la media de las coordenadas X e Y de los distintos expertos. Las unidades mostradas en los ejes representan la distancia existente entre los distintos expertos (que se obtiene de convertir las medidas de similitud, en este caso el porcentaje de acuerdo, en medidas de disimilitud). Sin embargo, lo verdaderamente importante del gráfico es que permite comparar las posiciones relativas de los distintos expertos según sus diagnósticos.

El botón del gráfico de burbujas permite que se superponga, sobre el gráfico MDS, los resultados del análisis cluster en forma de burbujas (Figura 8.32). Esto permite comparar de forma rápida y sencilla los resultados del análisis cluster y los resultados del MDS, y determinar cuál de ellos es el que presenta un mayor grado de similitud con las similitudes originales. En este caso el MDS presenta un mayor grado de correlación con los porcentajes de acuerdo iniciales.

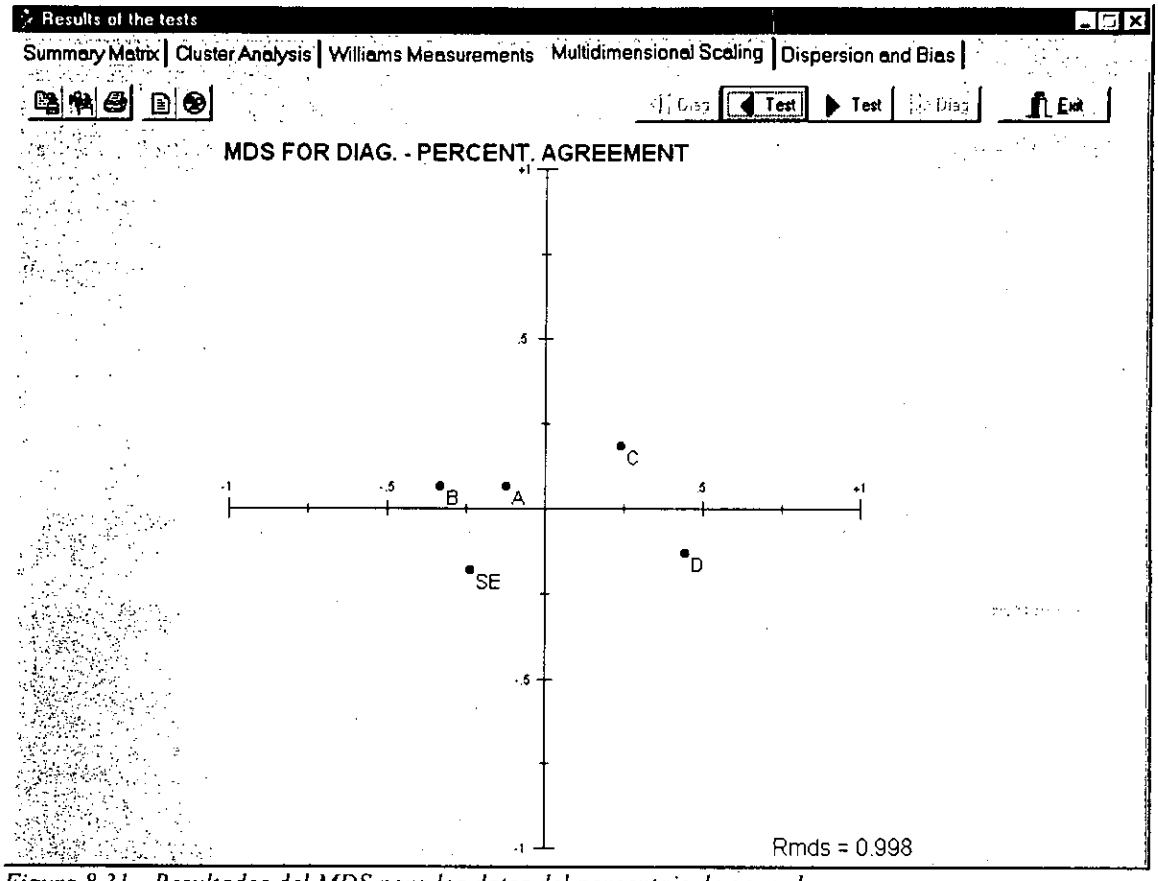


Figura 8.31 Resultados del MDS para los datos del porcentaje de acuerdo.

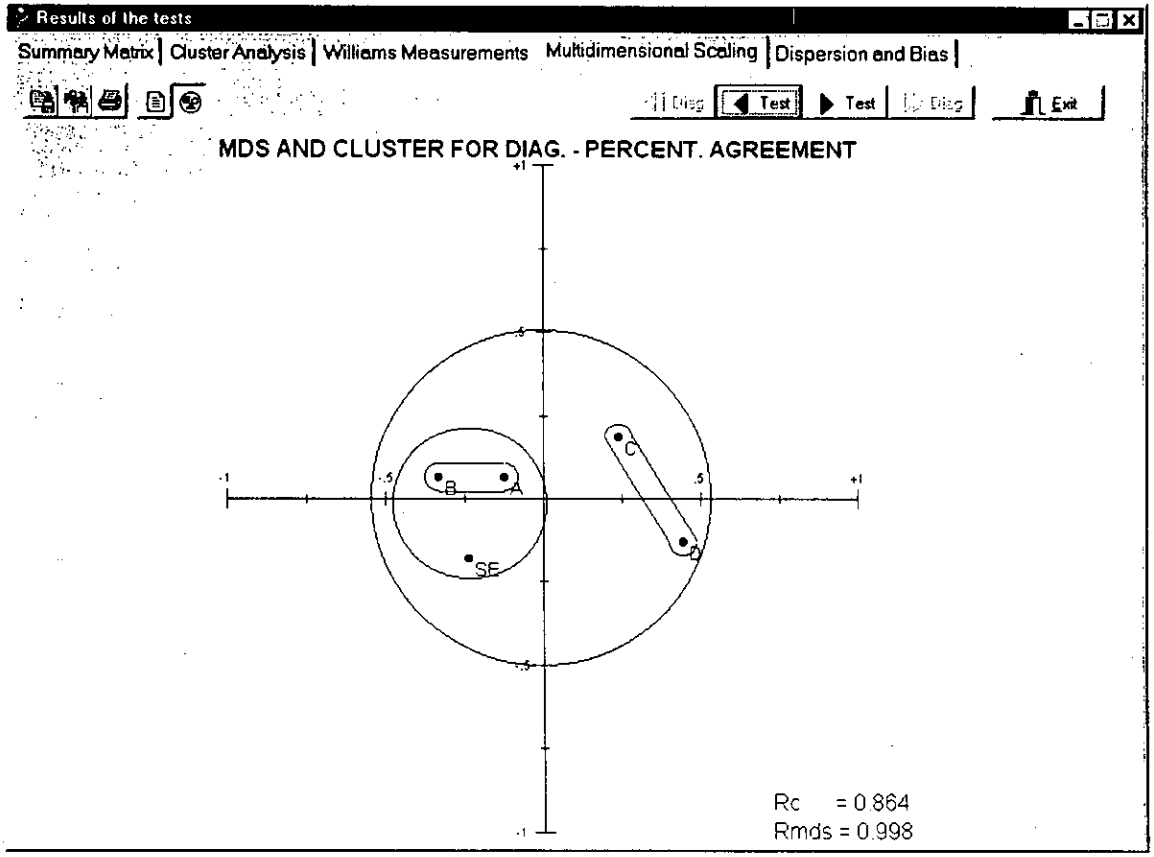


Figura 8.32 Resultados del MDS para los datos del porcentaje de acuerdo (incluyendo el análisis cluster como gráfico de burbujas).

El botón de la leyenda permite sustituir los nombres de los expertos por iconos con diversas figuras, e incluir una leyenda en la parte derecha del gráfico (Figura 8.33). Esta opción se incluyó por motivos de claridad ya que en ocasiones, los nombres de los expertos no se podían leer con facilidad al estar cruzados por muchas líneas.

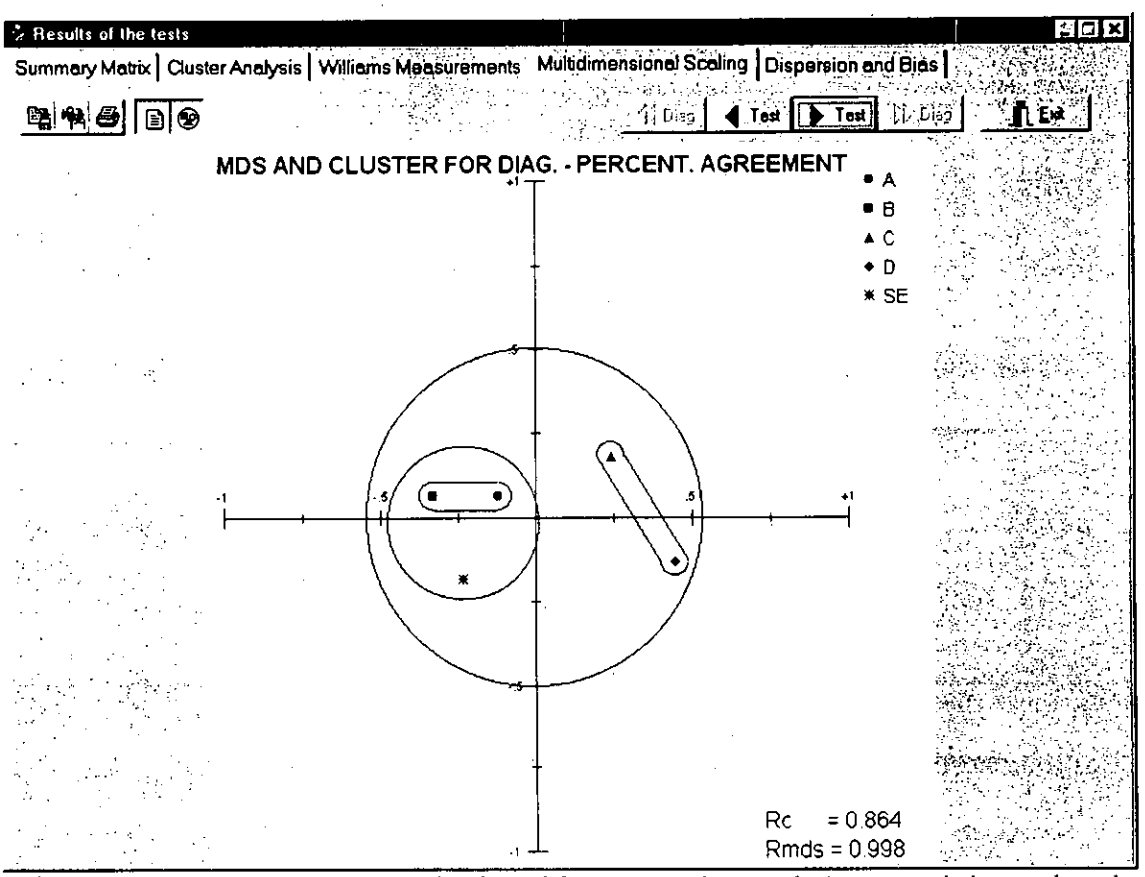


Figura 8.33 Resultados del MDS para los datos del porcentaje de acuerdo (sustituyendo los nombres de los expertos por iconos e incluyendo una leyenda en la parte derecha del gráfico).

Medidas de dispersión y tendencia

Las medidas de dispersión y tendencia son las únicas medidas de grupo que se obtienen directamente de la base de datos, y no a través de los tests de pares. Por dicha razón los botones que permiten cambiar entre los distintos tests están siempre desactivados en esta medida.

El gráfico que muestra los resultados de dispersión y tendencia es similar al gráfico de las matrices resumen y las medidas de Williams como puede verse en la Figura 8.34.

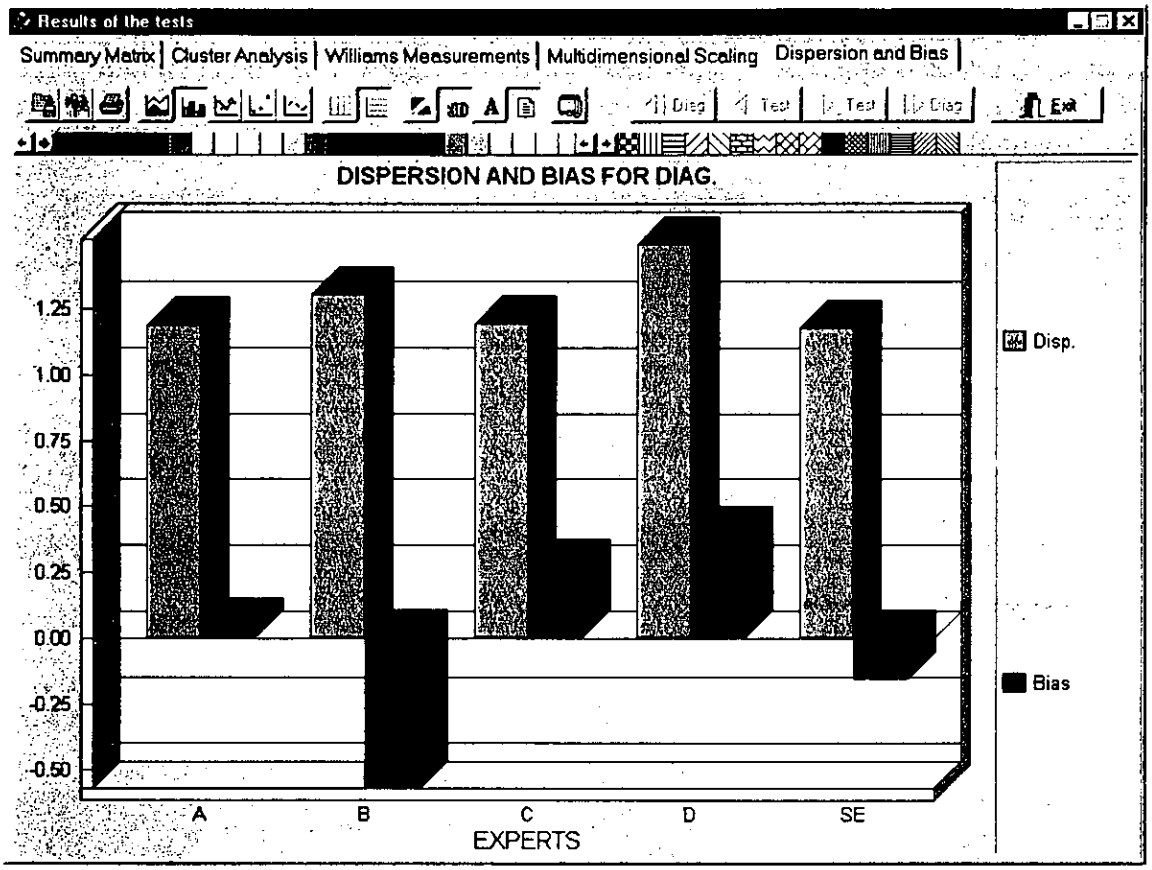


Figura 8.34 Medidas de dispersión y tendencia.

Resultados en formato texto

Hasta ahora hemos mostrado los resultados gráficos de las medidas de grupo. A pesar de su vistosidad, muchas veces es deseable mostrar los resultados en formato de texto, bien por cuestiones de espacio, o porque es necesario conocer de forma precisa los resultados de los tests.

Los resultados en modo texto se muestran en el mismo navegador utilizado por los tests de pares (Figura 8.24). Los resultados para el porcentaje de acuerdo del ejemplo seguido hasta el momento serían:

```
+-------------------------------------------------------------------+
| Diagnosis : DIAG.                                                 |
+-------------------------------------------------------------------+

****************** DISPERSION AND BIAS ********************

              DISP    BIAS
              -----   -----
Expert A    :  1.193   0.050
Expert B    :  1.306  -0.575
Expert C    :  1.198   0.275
Expert D    :  1.493   0.400
Expert SE   :  1.181  -0.150

+-------------------------------------------------------------------+
|                    Percent. agreement                             |
+-------------------------------------------------------------------+
```

```
******************** AGREEMENT MATRIX ***********************

      +---------------------------------+
      |   A     B     C     D     SE    |
+-----+---------------------------------+
| A   |  ----- 0.800 0.600 0.400 0.700  |
| B   |  0.800 ----- 0.400 0.200 0.700  |
| C   |  0.600 0.400 ----- 0.600 0.400  |
| D   |  0.400 0.200 0.600 ----- 0.300  |
| SE  |  0.700 0.700 0.400 0.300 -----  |
+-----+---------------------------------+

********************** CLUSTER ANALYSIS ***********************

Method        = Group Average
Rc            = 0.864
Num. clusters = 2

B - A         :  0.800
SE - B.A      :  0.700
D - C         :  0.600
--------------------------
SE.B.A - D.C :  0.383


******************* WILLIAMS MEASUREMENTS *******************

Expert A   :  1.442
Expert B   :  1.050
Expert C   :  0.968
Expert D   :  0.625
Expert SE  :  1.050


***************** MULTIDIMENSIONAL SCALING *****************

                X       Y
              -----   -----
Expert A   : -0.121   0.062
Expert B   : -0.330   0.064
Expert C   :  0.240   0.183
Expert D   :  0.447  -0.130
Expert SE  : -0.236  -0.182

Rmds =  0.998
```

8.4.4. Ratios de acuerdo

Los ratios de acuerdo tratan de medir el acuerdo existente entre un experto (o sistema experto) y una referencia estándar (que puede ser un experto de alto nivel, un consenso de expertos o la solución real del problema planteado).

Para el cálculo de los ratios de acuerdo es necesario acceder a la ventana correspondiente (Figura 8.35) a través del menú principal. En dicha ventana podemos seleccionar cuál es la referencia estándar, qué expertos vamos a comparar con ella, y qué diagnóstico y categoría vamos a tomar en consideración.

Además de los ratios de acuerdo podemos llevar a cabo dos medidas de similitud propias de las tablas 2 × 2 como son el porcentaje de acuerdo y la medida de Jaccard. De igual forma que en los otros tests los resultados pueden presentarse tanto en forma gráfica (Figura 8.36) como en formato texto (Figura 8.37).

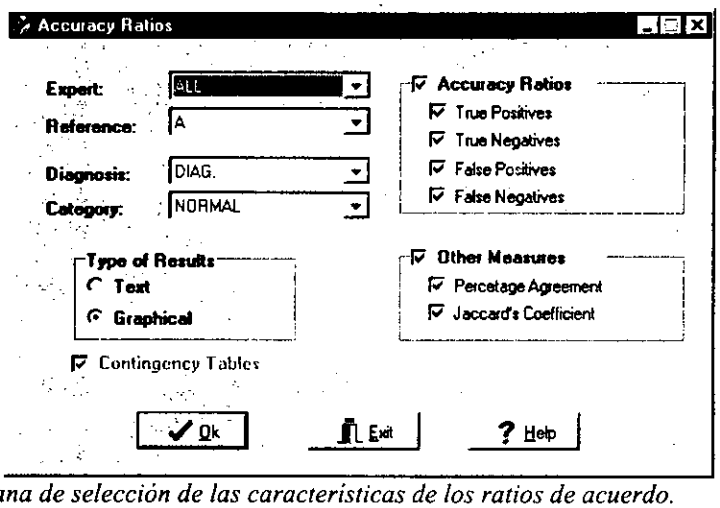


Figura 8.35 Ventana de selección de las características de los ratios de acuerdo.

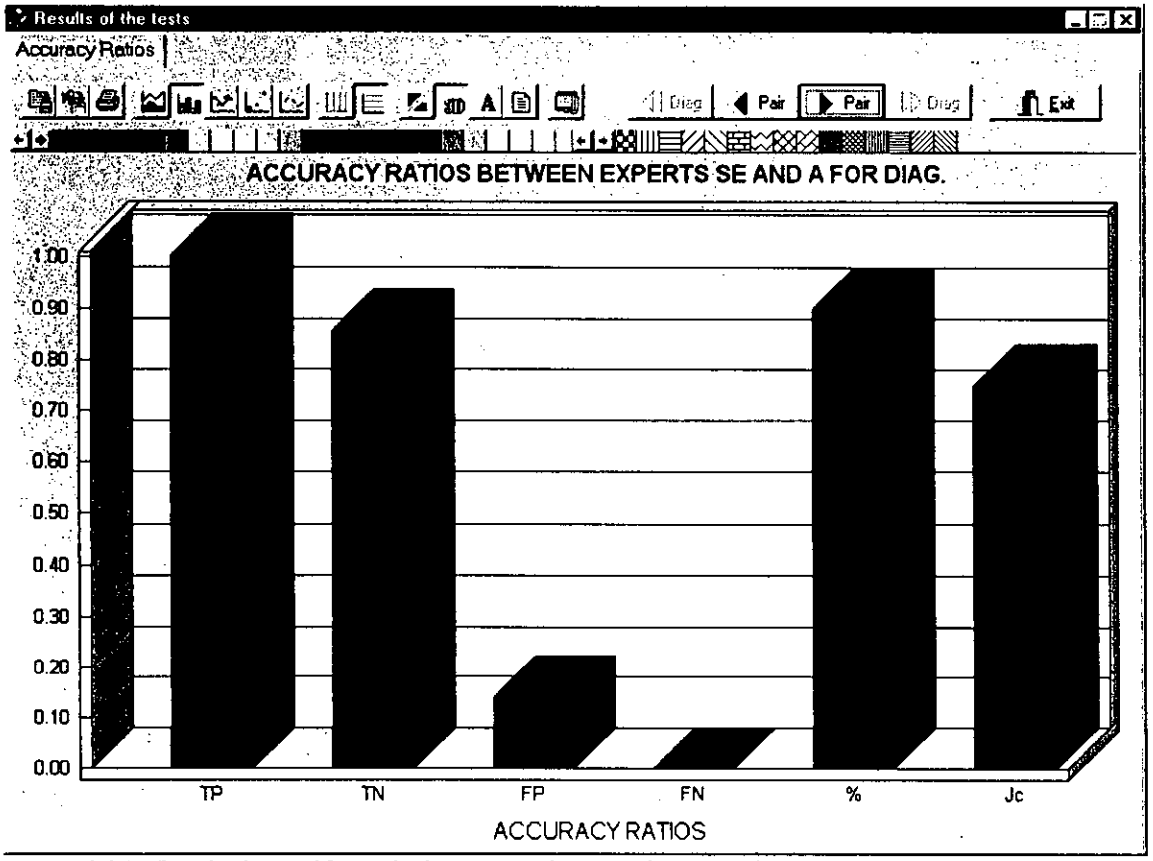


Figura 8.36 Resultados gráficos de los ratios de acuerdo para el sistema experto considerando al experto A como referencia estándar.

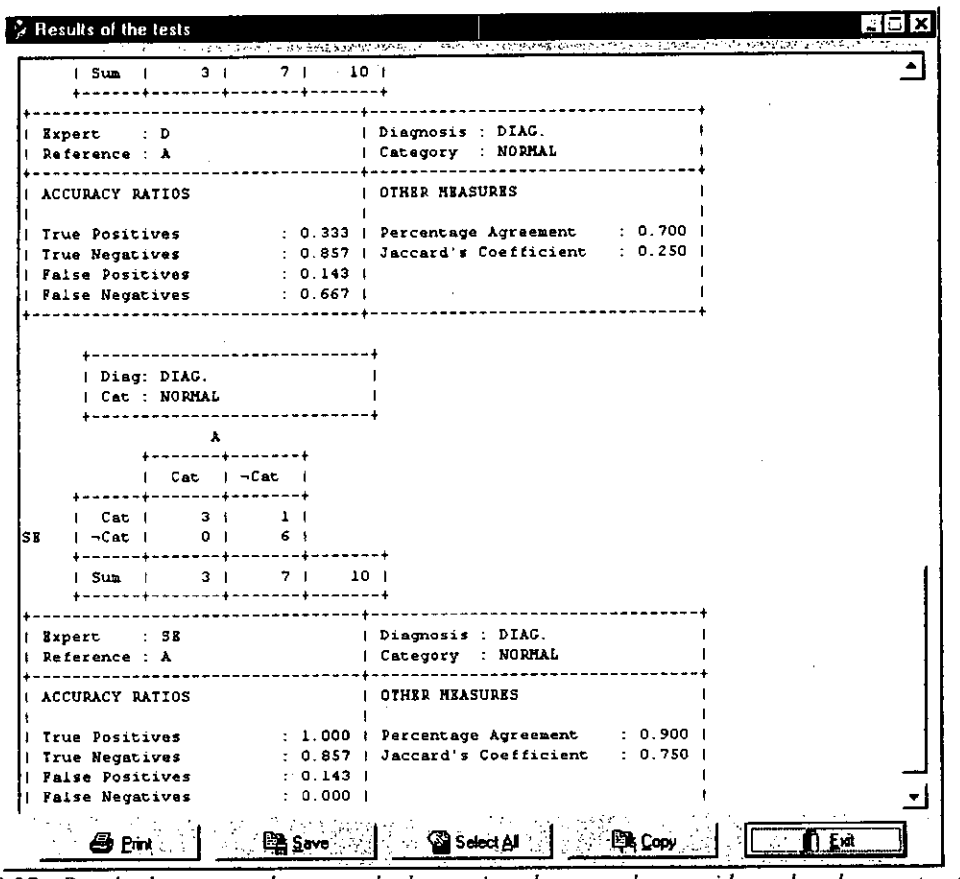


Figura 8.37 Resultados en modo texto de los ratios de acuerdo considerando al experto A como referencia estándar.

8.4.5. Menú de opciones

A la hora de validar un sistema experto puede ser necesario recurrir a SHIVA con frecuencia, bien porque se han obtenido nuevos datos, porque se ha probado una nueva modificación de la base de conocimientos o porque el sistema a validar engloba muchos módulos independientes. Generalmente, una vez que se ha desarrollado una estrategia de validación ésta permanece inalterable a través de los distintos ensayos.

Para simplificar su uso, SHIVA permite que se especifiquen las opciones que el usuario quiere que aparezcan por defecto cuando se abren las ventanas correspondientes a los distintos tests. Esto se hace a través del menú de opciones accesible desde el menú principal. Por ejemplo, en la Figura 8.38 puede verse que el usuario ha determinado que, por defecto, aparezcan seleccionadas las medidas de acuerdo, y no seleccionadas las medidas de asociación. Además los resultados se mostrarán en formato gráfico. Existen hojas similares para las medidas de grupo y los ratios de acuerdo.

En el menú de opciones también se puede acceder a la hoja "otras opciones" que se muestra en la Figura 8.39. En dicha hoja podemos cambiar el directorio por defecto en el que opera la herramienta, y el lenguaje por defecto en el que aparecen las distintas ventanas.

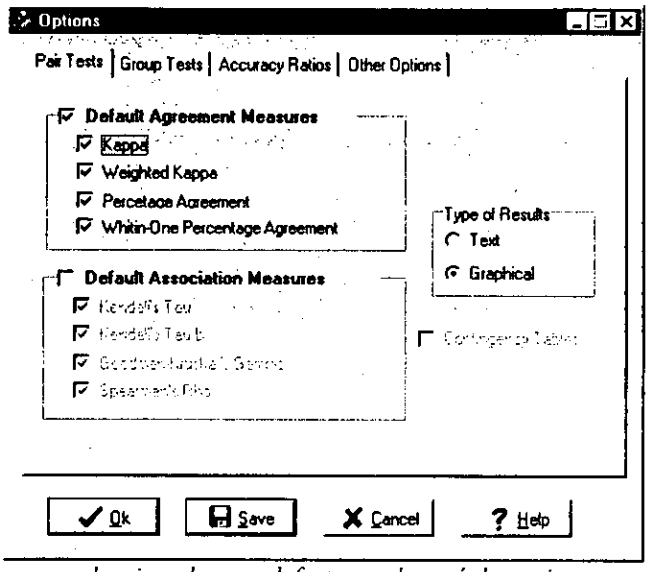


Figura 8.38 Tests de pares seleccionados por defecto en el menú de opciones.

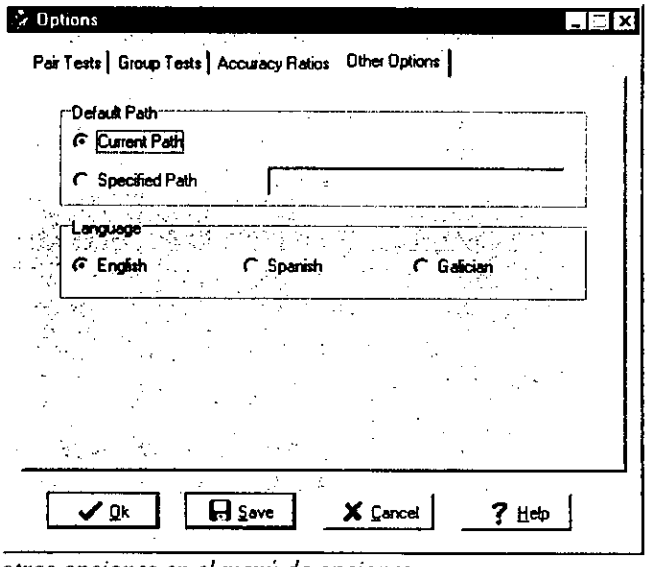


Figura 8.39 Hoja de otras opciones en el menú de opciones.

Las distintas opciones pueden utilizarse en la sesión en curso (pulsando el botón OK) o guardarse en un fichero para posteriores uso de la herramienta (pulsando el botón SAVE). El fichero en el que se guardan las distintas opciones es de tipo INI y un ejemplo del mismo sería el que se muestra a continuación (las opciones marcadas aparecen con un uno y las no marcadas con un cero; en caso de que sólo sea posible una opción de un grupo se indica el índice de dicha opción).

```
[Default Group Tests]
Agreement Matrix=1
Cluster Analysis=1
Cluster Algorithm=2
Williams Measurements=1
Multidimensional Scaling=1
Results=1
Join Pair Tests=1
Disp. and Bias=0

[Default Pair Tests]
Agreement Measures=1
Kappa=1
Weighted Kappa=1
Percentage Agreement=1
Within-one Percentage Agreement=1
```

```
Association Measures=0
Kendall's Tau=1
Kendall's Tau b=1
Goodman-Kruskal's Gamma=1
Spearman's Rho=1
Results=1
Contingency Tables=0

[Default Accuracy Ratios]
Accuracy Ratios=1
TP=1
TN=1
FP=1
FN=1
Other Coefficients=1
Percentage Agreement=1
Jaccard=1
Kappa=0
Results=0
Contingency Tables=1

[Other]
Default Path=0
Specified Path=c:\shiva
Language=1
```

8.4.6. Otras características

Además de las características reseñadas en los apartados anteriores, SHIVA incluye una serie de características adicionales como son:

- Ayuda del manejo de las distintas opciones del programa, así como de los distintos tests que implementa.
- Un instalador / desinstalador que permite su sencilla instalación en cualquier ordenador personal dotado del sistema operativo *Windows 95*.

8.5. Sistema experto de interpretación

De las tres fases de las que se compone nuestra metodología de validación (planificación, aplicación e interpretación), la más complicada es la última, ya que la interpretación de una medida es un proceso que no depende sólo del valor de la misma, sino también del contexto en que se encuentra. Así, que el sistema experto presente un porcentaje de acuerdo del 70% con los expertos puede ser aceptable en diversas situaciones, pero en otras se requerirá que dicho porcentaje no baje del 90%.

También es importante considerar el número de casos utilizados para la obtención de la medidas, la consistencia existente entre los propios expertos humanos, etc. Para ello es necesaria una considerable experiencia en la validación de sistemas expertos.

No obstante, ha sido planteada una aproximación en la que, por ejemplo, se desarrollen reglas que implementen la tabla interpretación del índice kappa desarrollada por Landis y Koch (1977) y que hemos visto en la Tabla 6.7. Así se han implementado estructuras de este tipo:

```
IF (kappa ≤ 0.00) THEN Nivel_de_acuerdo = Nulo
IF (kappa > 0.00) and (kappa ≤ 0.20) THEN Nivel_de_acuerdo = Insuficiente
IF (kappa > 0.20) and (kappa ≤ 0.40) THEN Nivel_de_acuerdo = Ligero
IF (kappa > 0.40) and (kappa ≤ 0.60) THEN Nivel_de_acuerdo = Moderado
IF (kappa > 0.60) and (kappa ≤ 0.80) THEN Nivel_de_acuerdo = Sustancial
IF (kappa > 0.80) and (kappa ≤ 1.00) THEN Nivel_de_acuerdo = Casi_perfecto o perfecto
```

A través del menú principal de SHIVA podemos acceder a la ventana principal de este sistema experto de interpretación (Figura 8.40). En dicha ventana seleccionamos el diagnóstico a examinar y la medida a utilizar (en este caso kappa o kappa ponderada).

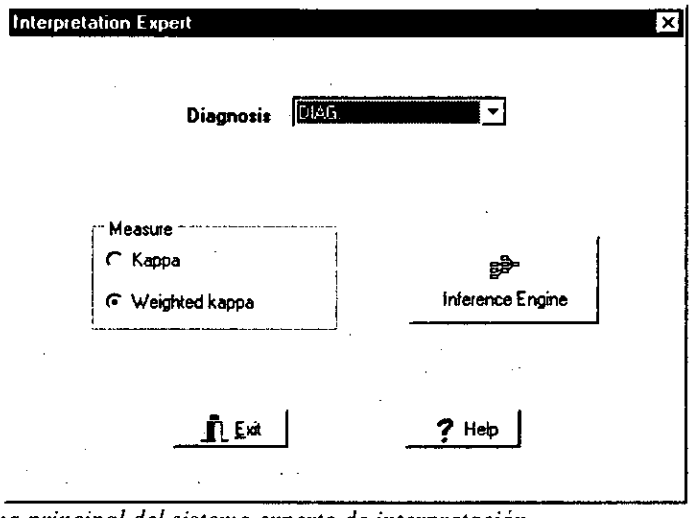


Figura 8.40 Ventana principal del sistema experto de interpretación

Ejecutando el motor de inferencias obtenemos los resultados que se muestran en la Figura 8.41.

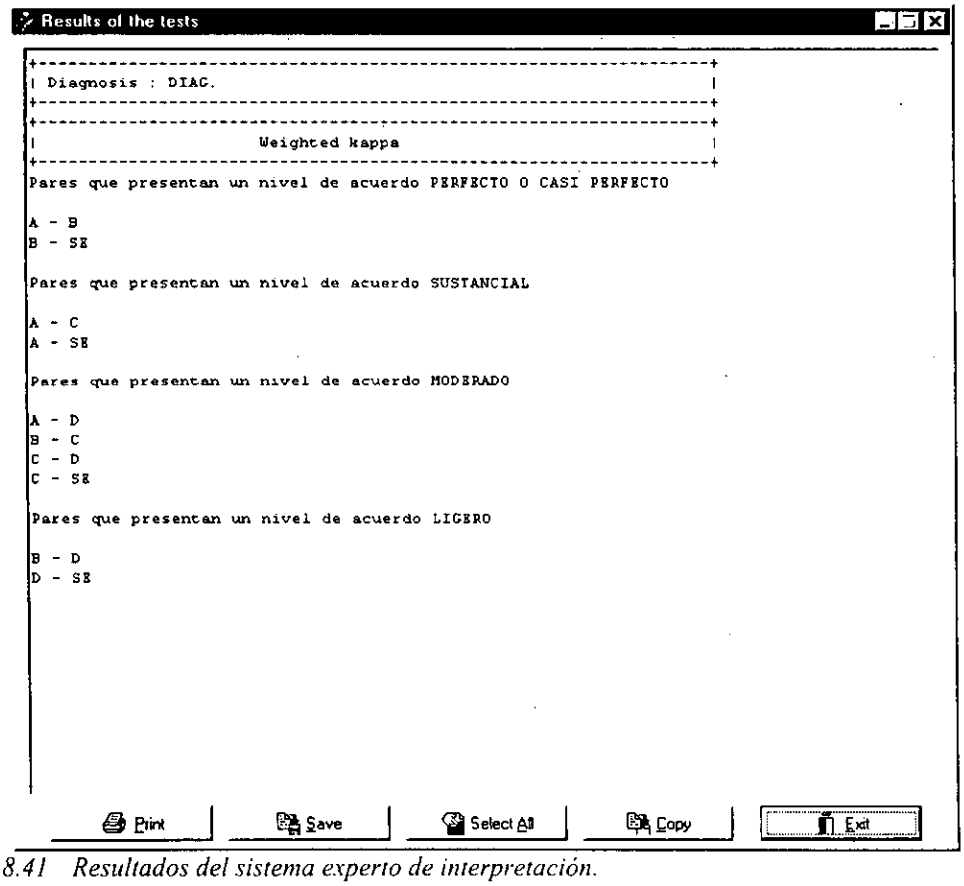


Figura 8.41 Resultados del sistema experto de interpretación.

En todo caso, la construcción de un sistema experto específico para la ayuda a la interpretación de los resultados obtenidos por SHIVA es un objetivo que, claramente, sobrepasa las pretensiones de esta tesis, y que constituye un línea en curso de investigación y desarrollo.

8.6. Resumen

En este capítulo hemos descrito la herramienta de validación SHIVA, desarrollada para facilitar la aplicación de la metodología propuesta de validación. SHIVA se ha desarrollado mediante la herramienta de programación Delphi y consta de 28 módulos que suman unas 14,000 líneas de código. El menú principal de SHIVA está constituido por una ventana, a través de la cual podemos acceder a los distintos módulos del sistema.

La primera fase de la metodología consiste en la planificación del proceso de validación. Esta fase puede llevarse a cabo en SHIVA a través del sistema experto de planificación, que realiza recomendaciones de validación a partir de las características del dominio de aplicación, del propio sistema experto y de la etapa de desarrollo en la que nos encontremos. El sistema experto de planificación ha sido desarrollado en la herramienta Nexpert Objtect, pero para su implementación en SHIVA se ha desarrollado un pequeño motor de inferencias que sigue una estrategia de búsqueda regresiva.

La siguiente fase de la metodología es la fase de aplicación. El primer paso de esta fase consiste en el preprocesado de la base de datos, que en SHIVA se realiza a través de un asistente. Este asistente facilita las labores de localización de la base de datos, identificación de su estructura, tipado de los campos, fijación del orden entre las categorías y establecimiento de los pesos de desacuerdo. La base de datos se almacena en ficheros dBase mientras que la información proveniente del preprocesado se almacena en ficheros de validación (VAL). SHIVA implementa un analizador sintáctico LL(1) para poder leer dichos ficheros de validación.

Después del preprocesado, se realiza la aplicación de las distintas medidas (tests de pares, de grupo y ratios de acuerdo). El resultado de estas medidas puede mostrarse en modo texto (a través del navegador de resultados de texto, también utilizado por el sistema experto de planificación) o en modo gráfico (a través del navegador de resultados gráficos).

La información mostrada por SHIVA como resultado de las distintas medidas estadísticas facilita la realización de la siguiente fase de la metodología (la fase de interpretación). También se ha diseñado un pequeño sistema experto que implementa las reglas de interpretación de Landis y Koch para las medidas kappa, dejándose como línea de trabajo futuro la realización de un sistema experto más elaborado.

A continuación veremos los resultados de aplicar la herramienta SHIVA a la validación de un par de sistemas expertos reales.

9. RESULTADOS

> Lo que muchos investigadores dan por descontado antes de empezar un experimento es infinitamente más interesante que cualquier resultado al que conduzca su experimento.
> *Norbert Wiener (Matemático estadounidense padre de la cibernética, 1874 – 1964).*
>
> No le digas a la gente cómo hacer las cosas. Dile lo que deben hacer y deja que te asombren con sus resultados.
> *George Patton (Militar estadounidense, 1885 – 1945).*

Una vez desarrolladas la metodología de validación y la herramienta que permite su aplicación, se procedió a investigar su comportamiento para tratar de evaluar su aplicabilidad a sistemas reales. Para ello se utilizó la herramienta SHIVA en la validación de los sistemas inteligentes PATRICIA (Moret et al., 1993) y NST-EXPERT (Alonso et al., 1995), desarrollados en el Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA) de la Universidad de A Coruña.

Es importante destacar que, los resultados que se van a exponer a continuación se refieren a la herramienta SHIVA y a la metodología de validación que subyace. Los sistemas expertos PATRICIA y NST-EXPERT se utilizan aquí como elementos de experimentación, dadas sus características particulares, por lo que tales resultados no pretenden reflejar el comportamiento de dichos sistemas.

9.1. Resultados de la aplicación de SHIVA sobre el sistema de monitorización inteligente PATRICIA

Desde la perspectiva de SHIVA, PATRICIA se caracteriza por:

- Estar diseñado para asistir a los médicos en el manejo de pacientes, lo que define un dominio crítico.
- La arquitectura del sistema se divide en siete módulos independientes entre si, pero relacionados a través de sus respectivas entradas y salidas.
- No existe una solución real contra la cual comparar los resultados del sistema, por lo que utilizaremos como referencia un grupo de expertos del domino.
- Los usuarios del sistema son expertos humanos del dominio.
- Las entradas al sistema son una serie de parámetros, una serie de contextos naturales (que afectan al procesado simbólico de los parámetros), y unos contextos inferenciales (que pueden influir en el establecimiento de las distintas interpretaciones).
- Las interpretaciones del sistema son clasificadas en categorías diagnósticas y terapéuticas, y se matizan con una serie de etiquetas lingüísticas que siguen un orden jerárquico.
- PATRICIA no utiliza medidas de incertidumbre, aunque para la clasificación de algunos parámetros se utiliza cuantificación difusa.
- El sistema ha sido diseñado para ser integrado en un sistema de información más amplio (en este caso un HIS, Hospital Information System).

- PATRICIA es un prototipo de campo validado.

En base a las características de PATRICIA podemos llevar a cabo su validación siguiendo las fases de la metodología propuesta en el capítulo 7 de este trabajo.

9.1.1. Fase de planificación

En base a las características del dominio, del sistema y de la fase de desarrollo el planificado de SHIVA define las siguientes características de la estrategia de validación:

- Relativas al dominio:
  - * Al ser un dominio crítico es necesario realizar un validación rigurosa, ya que los errores del sistema pueden tener consecuencias inaceptables.
  - * Al no haber una salida real con la que comparar los resultados no se puede llevar a cabo una validación contra el problema. Por lo tanto nos vemos limitados a llevar a cabo solamente una validación contra el experto.
  - * La existencia de un grupo de expertos favorece la objetividad del estudio pero pueden surgir problemas si las discrepancias entre los expertos son elevadas. En este caso se sugeriría elaborar un consenso entre los distintos expertos.
  - * Podemos llevar a cabo medidas de pares entre los distintos expertos y el sistema experto, y medidas de grupo a partir de la información suministrada por dichas medidas de pares. Los ratios de acuerdo no se pueden realizar al no existir una referencia estándar.
  - * Los usuarios del sistema también pueden colaborar en la validación a través de tests de campo. Aunque esta posibilidad queda limitada al tratarse de un dominio crítico.
- Relativas al sistema:
  - * Al ser un sistema de carácter modular podemos realizar la validación de cada uno de los módulos de forma independiente.
  - * Cinco módulos incluyen categorías diagnósticas que pueden validarse en base a casos históricos o con casos actuales.
  - * Dos módulos incluyen categorías terapéuticas que sólo pueden validarse de forma retrospectiva, comparando las terapias del sistema con las terapias propuestas por los expertos humanos. Al ser un dominio crítico no se pueden validar estos módulos de forma prospectiva (es decir, aplicar la terapia y ver si su evolución es correcta) a no ser que coincidan retrospectivamente con los expertos humanos.
  - * Como los resultados son de tipo ordinal, es necesario tener en cuenta la distancia existente entre las categorías a la hora de evaluar el impacto de las posibles discrepancias. Por ello se recomienda la utilización de medidas como kappa ponderada, porcentajes de acuerdo dentro de uno y medidas de asociación.

* Mientras el sistema no esté integrado en un HIS, la validación puede ejecutarse de manera independiente. Tras su integración habrá que validar también los interfaces de comunicación.

- Relativas a la fase de desarrollo:

  * Al tratarse de un prototipo de campo, la validación debe ser más rigurosa que en fases iniciáticas, con casos que abarquen toda la cobertura posible, y con expertos ajenos al desarrollo del sistema.

9.1.2. Fase de aplicación

La fase de aplicación de la metodologia de validación propuesta se compone a su vez de otras subfases como son la captura de la casuística, el preprocesado de los datos y la realización de las medidas estadísticas (en las que incluimos las medidas de pares, las medidas de grupo y los ratios de acuerdo).

Captura de la casuística

Para la captura de la casuística se utilizaron los formularios de elicitación-validación, empleados también en la adquisición del conocimiento de los expertos humanos. Estos formularios se dividen en dos partes: (a) información de las características del caso tomado en consideración, y (b) interpretaciones de los expertos en base a la información suministrada. Las características de la casuística utilizada en la validación se resumen en la Tabla 9.1.

Característica	Valor
Casos reales y representativos	30
Experto humanos	6
Categorías diagnósticas	5
Categorías terapéuticas	2
Datos numéricos totales	1470
Identificación de los expertos	letras A, B, C, D, E y F
Identificación de PATRICIA	letra G

Tabla 9.1 Características de la casuística de validación de PATRICIA.

Preprocesado de los datos

Una vez capturada la casuística se llevo a cabo un preprocesado de los datos en el que se efectuaron las siguientes acciones:

- Corrección de errores.

- Trasvase de los datos a un formato electrónico para ser fácilmente manipulables por el computador (formato dBase).

- Inclusión de información adicional en la base de datos de validación. Información que es recogida en un fichero de tipo VAL y que incluye la estructura de la base de datos, la identificación de los campos de la base de datos, el orden de las categorías semánticas y la ponderación de las discrepancias obtenidas.

Realización de las medidas estadísticas (tests de pares y tests de grupo)

A continuación describiremos los resultados obtenidos tras la aplicación de las distintas medidas estadísticas a algunos módulos del sistema.

Sea el modulo diagnóstico D1, en el que las interpretaciones se dividen en cinco categorías semánticas ordenadas. Para este caso se realizó una ponderación que seguía un progresión geométrica, de forma que una discrepancia en categorías muy alejadas es mucho más penalizada que una discrepancia en categorías adyacentes (Tabla 9.2).

		Experto B				
		1	2	3	4	5
Experto A	1	0	1	4	9	16
	2	1	0	1	4	9
	3	4	1	0	1	4
	4	9	4	1	0	1
	5	16	9	4	1	0

Tabla 9.2 *Ponderación de los distintos desacuerdos para la categoría diagnóstica D1*

Los resultados de los tests de pares pueden verse en la Figura 9.1. En este caso hemos decidido utilizar las siguientes medidas de pares:

- Porcentaje de acuerdo. Por ser una de las medidas más populares a la hora de medir acuerdos.
- Kappa ponderada. Porque permite eliminar los acuerdos debidos a la casualidad y ponderar las discrepancias según su importacia.
- Porcentaje de acuerdo dentro de uno. Porque permite considerar como acuerdos parciales aquellos diagnósticos que sólo se diferencian en una única etiqueta lingüística.
- Rho de Spearman. Porque nos permite analizar las asociaciones, no sólo los acuerdos, y su forma de tratar las ligaduras es más adecuada que la de otras medidas de asociación (para entornos de validación).

Como vemos en la Figura 9.1 existe un alto grado de concordancia. Los valores del porcentaje de acuerdo dentro de uno y de la rho de Spearman se hallan siempre por encima del 0.9, y los valores de kappa ponderada y el porcentaje de acuerdo por encima del 0.8.

Los menores valores aparecen en los pares A-B, A-C, B-C, B-D, C-G y D-G.

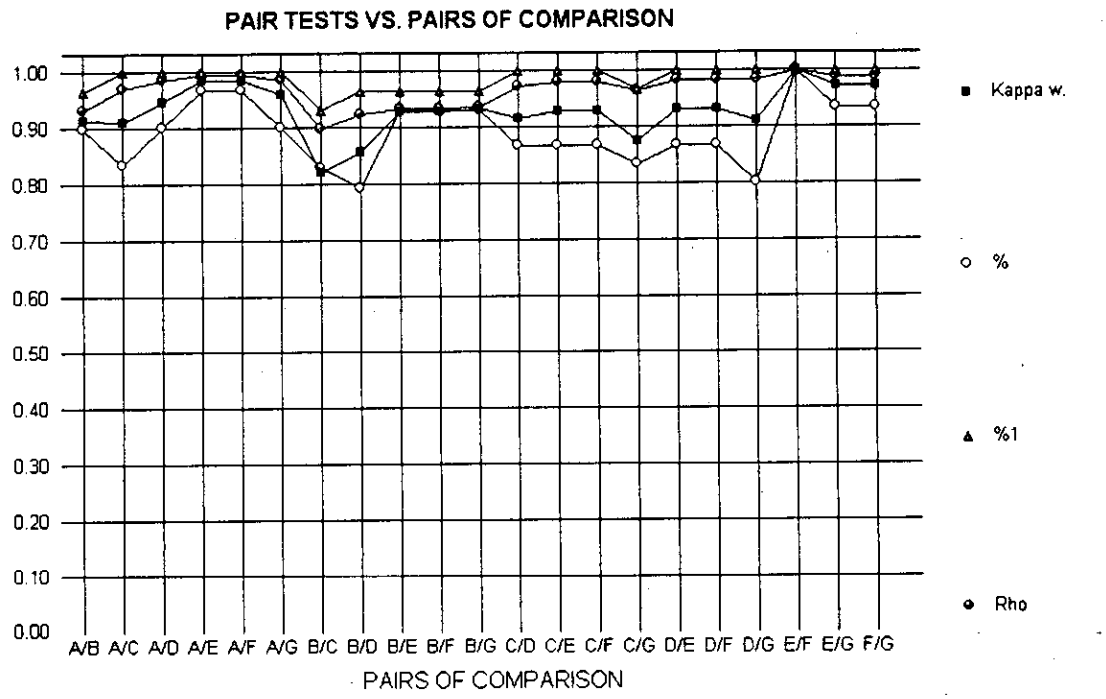


Figura 9.1 Resultados de los tests de pares para la categoría diagnóstica D1

En base a los valores obtenidos por los tests de pares desarrollamos las medidas de grupo. En la Figura 9.2 se muestran los resultados de las medidas de Williams.

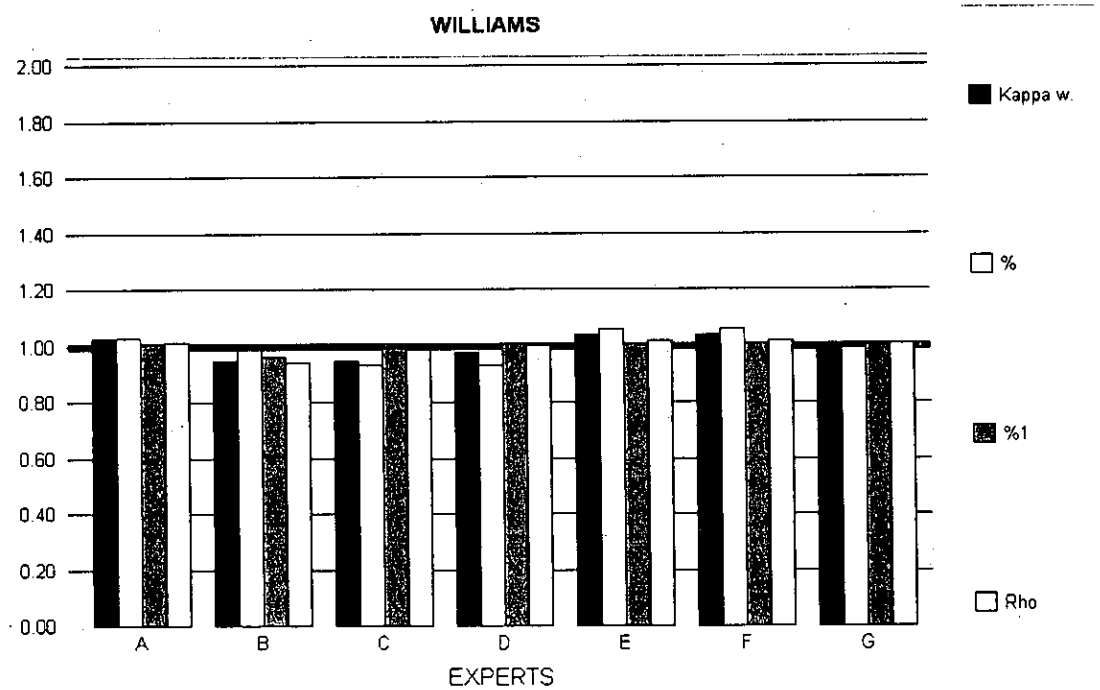


Figura 9.2 Resultados de los tests de pares para la categoría diagnóstica D1

Como podemos ver para el porcentaje de acuerdo dentro de uno y para rho los resultados son prácticamente la unidad para casi todos los expertos. Sin embargo para kappa ponderada y para el porcentaje de acuerdo los resultados de B, C y D son inferiores a la unidad.

Los resultados para el análisis cluster se muestran en la Figura 9.3. Incluimos sólo los resultados de kappa ponderada y el porcentaje de acuerdo porque para el porcentaje de acuerdo dentro de uno y para la rho de Spearman las diferencias son mínimas, y casi imperceptibles en el gráfico.

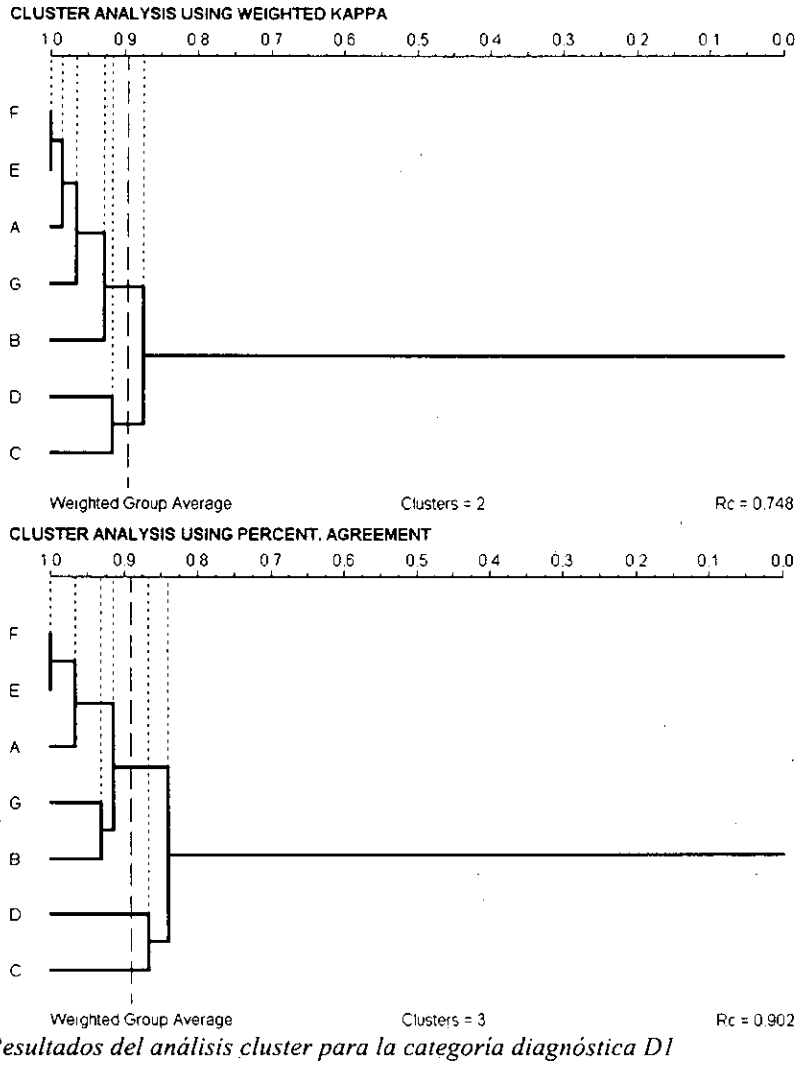


Figura 9.3 Resultados del análisis cluster para la categoría diagnóstica D1

Como vemos, los resultados de E y F son casi idénticos, después se le agrupan los resultados de A y G, y por último se añaden los resultados de B, C y D (el orden en que se añaden estos últimos expertos depende de la medida que estemos considerando).

Los resultados del escalamiento multidimensional se muestran en la Figura 9.4. En ella podemos ver como los resultados de A, E y F son casi idénticos. Junto a ellos se sitúan los resultados de G y después los resultados de B, C y D. Los resultados de B se hallan cerca de G mientras que los resultados de C y D se encuentran al otro lado del gráfico (recordemos que la disposición de los ejes es arbitraria).

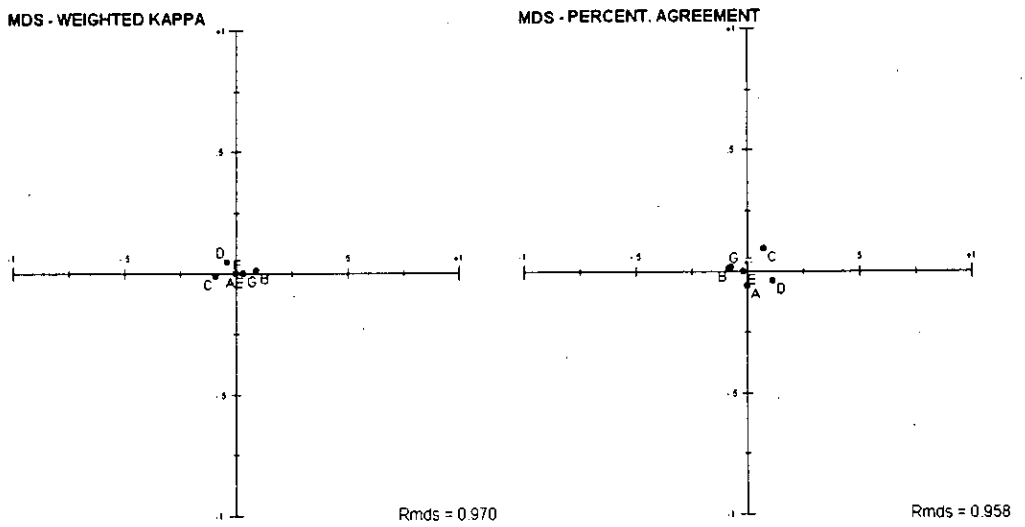


Figura 9.4 Resultados del análisis cluster para el diagnóstico D1

Por último, los datos de dispersión y tendencias se muestran en la Figura 9.5. Como vemos los expertos que presentan una mayor dispersión en sus diagnósticos son B, C y D (aunque la tendencia de los resultados de B es distinta a la de los resultados de C y D).

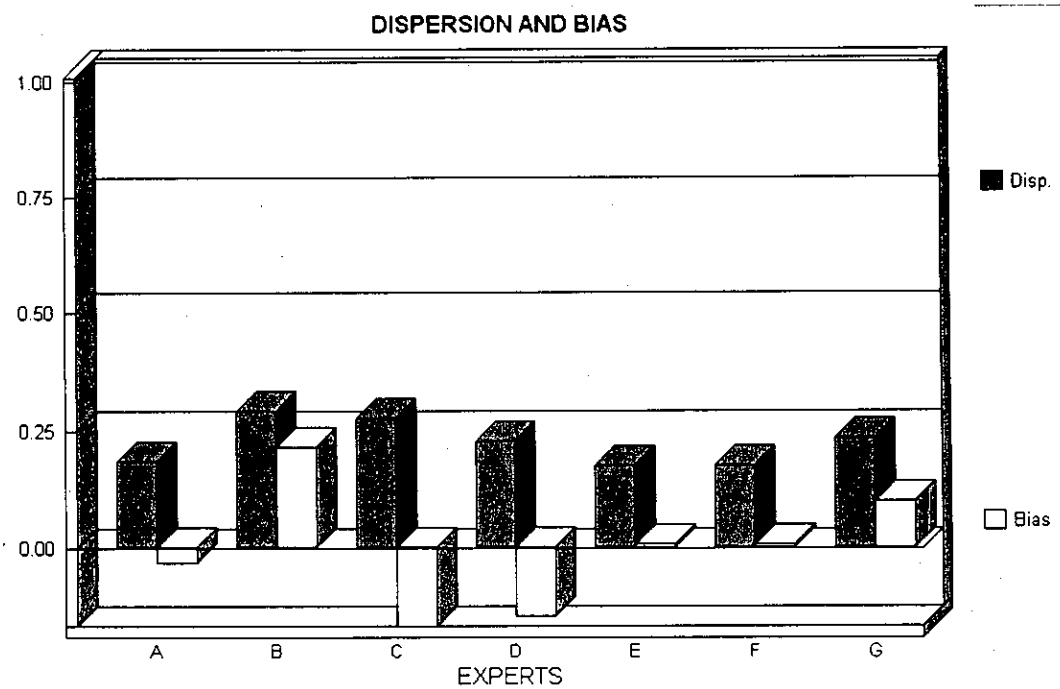


Figura 9.5 Datos de dispersión y tendencia para el diagnóstico D1

En la Figura 9.6 representamos los resultados para la categoría diagnóstica D2 y en la Figura 9.7 los resultados para la categoría diagnóstica D3. Como vemos, en ambos los resultados del sistema experto (G) se sitúan fuera del grupo principal de expertos.

A modo de resumen mostramos sólo los resultados del MDS con las burbujas del análisis cluster superpuesto, para los tests de kappa ponderada y el porcentaje de acuerdo.

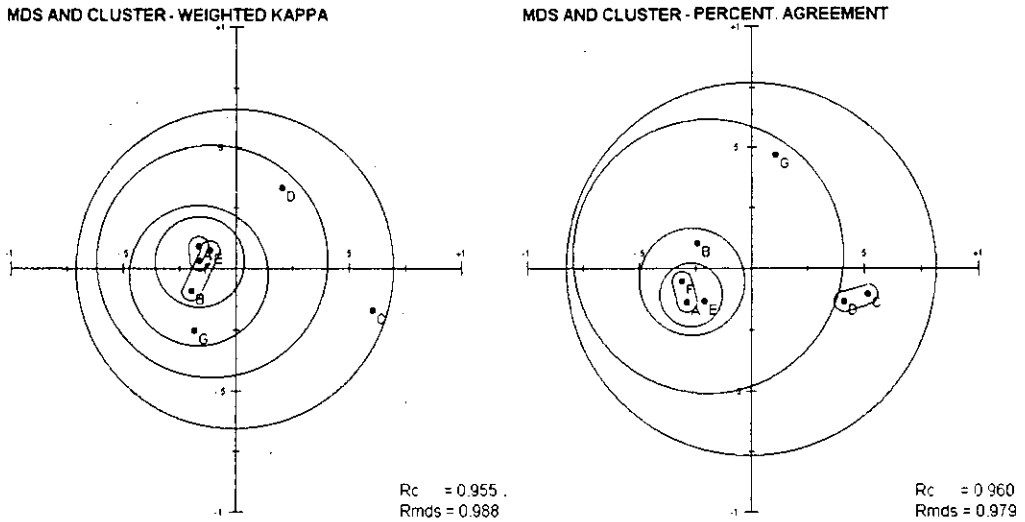


Figura 9.6 Resultados de la categoría diagnóstica D2.

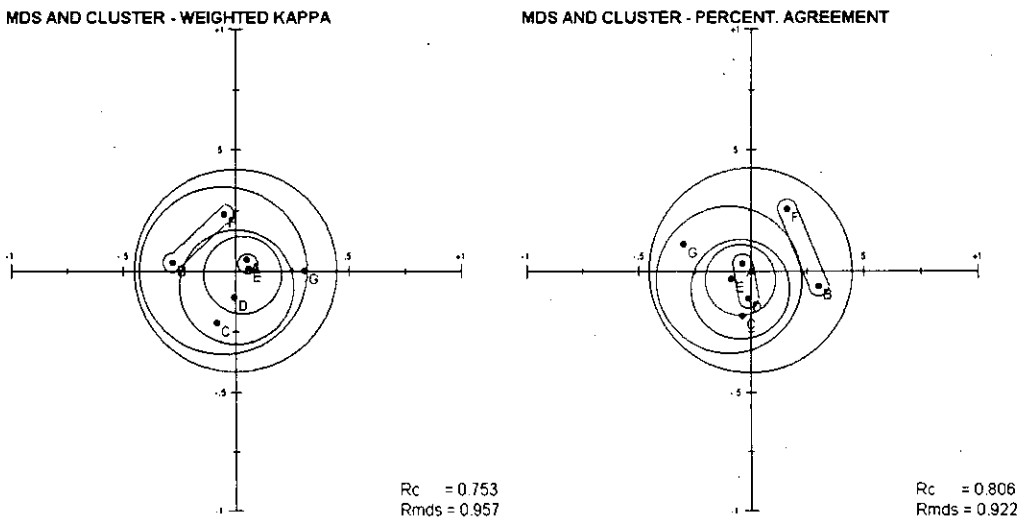


Figura 9.7 Resultados de la categoría diagnóstica D3.

Es importante destacar que la categoría diagnóstica D3 se diferencia de otras categorías diagnósticas del sistema en que la relación existente entre las distintas categorías tiene un orden, pero este orden no tiene por qué ser lineal. Esto obligó a realizar un profundo estudio en el que se pudieran determinar los pesos adecuados para las distintas discrepancias, los resultados de este estudio se pueden ver en la Tabla 9.3.

	1	2	3	4	5	6	7	8	9	10	11
1	0	3	3	3	3	1	1	1	1	4	4
2	3	0	1	6	6	1	2	5	4	1	7
3	3	1	0	6	6	2	1	4	5	1	7
4	3	6	6	0	1	5	4	1	2	7	1
5	3	6	6	1	0	4	5	2	1	7	1
6	1	1	2	5	4	0	1	3	2	3	6
7	1	2	1	4	5	1	0	2	3	3	6
8	1	5	4	1	2	3	2	0	1	6	3
9	1	4	5	2	1	2	3	1	0	6	3
10	4	1	1	7	7	3	3	6	6	0	8
11	4	7	7	1	1	6	6	3	3	8	0

Tabla 9.3 Pesos para las discrepancias entre los posibles resultados de la categoría diagnóstica D3

Los resultados obtenidos para la categoría diagnóstica D2 se utilizan como información de entrada para el módulo de terapia T2. De la misma forma, los resultados

de D3 son utilizados por T3. Si analizamos los resultados de las terapias vemos que los resultados del sistema experto han mejorado y se sitúan más cerca del grupo principal de expertos (como se muestra en la Figura 9.8 y en la Figura 9.9). La explicación de estos resultados se verá en la fase de interpretación.

También, por norma general, podemos ver que los resultados del MDS son más representativos que los resultados del clustering porque su grado de correlación con las similitudes originales es más elevado.

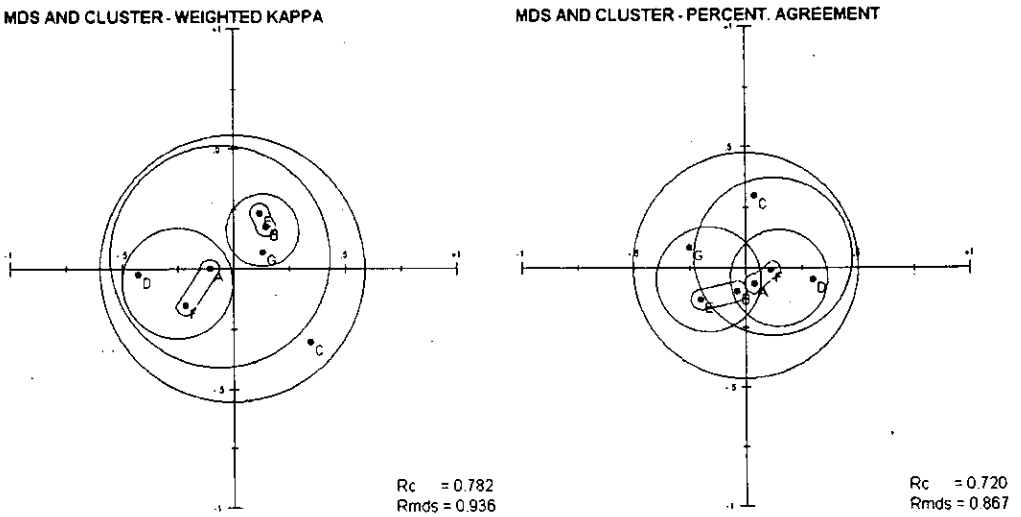


Figura 9.8 Resultados de la terapia T2.

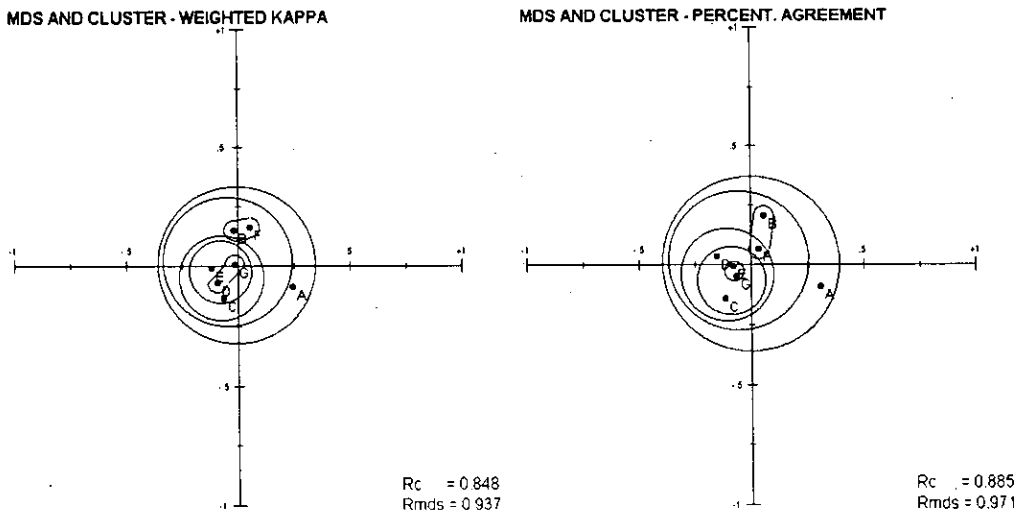


Figura 9.9 Resultados de la terapia T3.

9.1.3. Fase de interpretación

En esta fase se interpretan los resultados obtenidos en la fase anterior. Si tomamos en consideración la categoría diagnóstica D1 podemos ver que los resultados del sistema experto son excelentes, ya que se sitúa a nivelen muy similares a los de los expertos. Así, por ejemplo, vemos que el valor de kappa ponderada rara vez baja de 0.9 lo que, según las reglas de interpretación de Landis y Koch, es un acuerdo casi perfecto.

También vemos que para ese diagnóstico los peores resultados son los obtenidos por B, C y D. Además los resultados de B son bastante diferentes a los de C y D.

En cuanto a las categorías diagnósticas D2 y D3, y a las categorías terapéuticas T2 y T3, la razón de que los resultados de las terapias sean mejores que los resultados de los diagnósticos indica que, a la hora de establecer las diagnósticos, los expertos y el sistema experto no siguen los mismos criterios. Sin embargo, la información que no se considera en el diagnóstico sí se considera en la terapia y las conclusiones de los expertos y el sistema experto son más coincidentes.

En el caso de la categoría diagnóstica D2 se descubrió que uno de los parámetros que el sistema experto utilizaba para establecer la terapia T2 era utilizado por los expertos en el diagnóstico. Esto provocaba situaciones curiosas como la que se muestra en la Tabla 9.4, en la que los diagnósticos de los expertos A, E y G son distintos pero sus terapias son prácticamente idénticas.

D2			T2					
A	E	G	A		E		G	
↑	=	↑↑	↓	0.90	↓	0.90	↓	0.87
↑	=	↑↑	↓	0.95	↓	0.80	↓	0.92

Tabla 9.4 Comparación entre el diagnóstico D2 y la terapia T2.

Para el caso de la categoría diagnóstica D3, el sistema experto modificaba el valor del diagnóstico según el contexto en el que se encontrara (por ejemplo, no es lo mismo la frecuencia cardíaca de un deportista que la de un fumador sedentario). Sin embargo los expertos no utilizaban el contexto a la hora de realizar el diagnóstico sino a la hora de realizar la terapia (tal y como se muestra en la Figura 9.10).

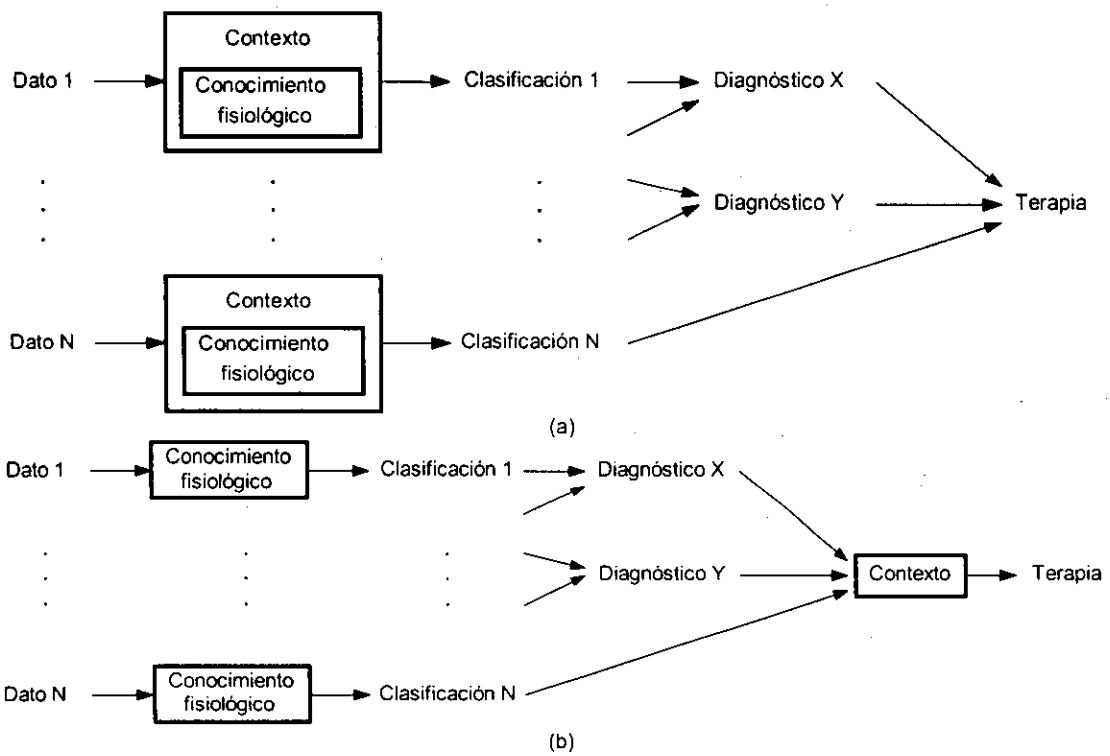


Figura 9.10 (a) Proceso de diagnóstico y terapia seguido por el sistema experto, (b) proceso de diagnóstico y terapia seguido por los expertos.

Si eliminamos el contexto en la realización del diagnóstico vemos que los resultados del sistema experto mejoran considerablemente y son prácticamente idénticos a los de los expertos humanos (Figura 9.11).

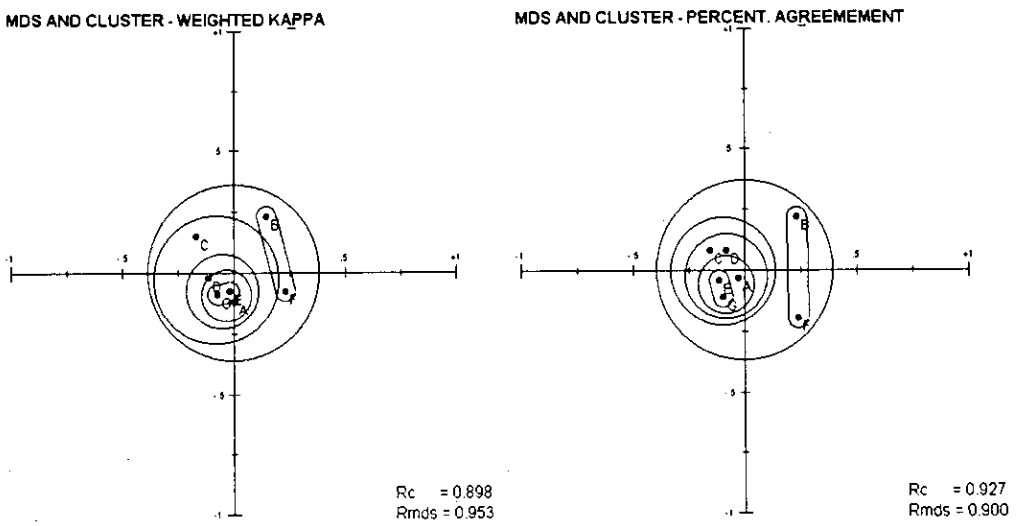


Figura 9.11 Datos de la categoría diagnóstica D3 obtenidos sin considerar los contextos.

Como vemos el proceso de validación de un sistema no sólo sirve para comprobar que un sistema funciona correctamente, sino que también sirva para adquirir nuevo conocimiento o refinar el ya existente.

9.2. Resultados de la aplicación de SHIVA al sistema experto NST-EXPERT

Desde la perspectiva de SHIVA, NST-EXPERT se caracteriza por:

- Estar diseñado para asistir a los médicos en el manejo de pacientes, lo que define un dominio crítico.
- La arquitectura del sistema se divide en 3 módulos independientes entre si, pero relacionados a través de sus respectivas entradas y salidas. Estos módulos corresponden, respectivamente, a categorías diagnósticas, de pronóstico y de terapia. Sin embargo este estudio se centró exclusivamente en el módulo de pronóstico, para el cual se desarrollaron nuevas aproximaciones.
- El módulo de pronóstico puede validarse contra los pronósticos de los expertos o contra la solución real de dicho pronóstico.
- Los usuarios del sistema son expertos humanos del dominio.
- La entrada al sistema lo constituyen una serie de evidencias, a partir de las cuales se puede desarrollar el pronóstico.
- Las interpretaciones del sistema y los expertos se matizan con una serie de etiquetas lingüísticas que siguen un orden jerárquico.
- El sistema utiliza medidas de incertidumbre.
- El sistema ha sido diseñado para ser integrado en un sistema de información más amplio (en este caso un HIS).
- NST-EXPERT es un prototipo de campo validado.

Para realizar la validación de este sistema seguimos las fases de planificación, aplicación e interpretación incluidas en la metodología propuesta.

9.2.1. Fase de planificación

En base a las características del dominio, del sistema y de la fase de desarrollo el planificador de SHIVA define las siguientes características de la estrategia de validación:

- Relativas al dominio:
  - * Al ser un dominio crítico es necesario realizar un validación rigurosa, ya que los errores del sistema pueden tener consecuencias inaceptables.
  - * La validación puede realizarse contra el experto o contra el problema. Si los resultados del sistema experto difieren de los resultados de los expertos, pero son similares a los de la solución real del problema, la confianza en el sistema aumentará.
  - * Pueden utilizarse medidas de pares y medidas de grupo (con los expertos humanos, el sistema experto y la salida real) y ratios de acuerdo (tomando como referencia la salida real).
  - * Los usuarios del sistema también pueden colaborar en la validación a través de tests de campo. Aunque esta posibilidad queda limitada al tratarse de un dominio crítico.
- Relativas al sistema:
  - * Al ser un sistema de carácter modular, podemos realizar la validación de cada uno de los módulos de forma independiente. En este caso nos centramos únicamente en la validación del módulo de pronóstico.
  - * El módulo de pronóstico puede validarse en base a casos históricos o con casos actuales.
  - * Se utilizan distintas aproximaciones en la construcción del módulo de pronóstico (factores de certeza, teorema de Bayes, análisis discriminante y redes de neuronas artificiales). El objetivo de la validación será dilucidar cual de las distintas aproximaciones presenta mejor rendimiento.
  - * El resultado del pronóstico puede ser "bueno" o "malo". Los resultados malos aparecen cuantificados por una medida de incertidumbre, para que dichos resultados puedan ser tratados por SHIVA se agrupan en tres etiquetas semánticas (ligeramente malo, moderadamente malo, bastante malo).
  - * Como los resultados son de tipo ordinal, es necesario tener en cuenta la distancia existente entre las categorías a la hora de evaluar el impacto de las posibles discrepancias. Por ello se recomienda la utilización de medidas como kappa ponderada, porcentajes de acuerdo dentro de uno y medidas de asociación.

* Mientras el sistema no esté integrado en un HIS, la validación puede ejecutarse de manera independiente. Tras su integración habrá que validar también los interfaces de comunicación.

- Relativas a la fase de desarrollo:

  * Al tratarse de un prototipo de campo, la validación debe ser más rigurosa que en fases iniciáticas, con casos que abarquen toda la cobertura posible, y con expertos ajenos al desarrollo del sistema.

9.2.2. Fase de aplicación

Captura y preprocesado de los datos

En este estudio se disponía de dos bases de datos. La primera de ellas contiene 3209 casos en los que se incluían las evidencias necesarias para establecer un diagnóstico y el resultado final del pronóstico. Esta base de datos se utilizó para entrenar los distintos modelos implementados en el módulo de pronóstico. La segunda base de datos contenía 177 casos con la misma información que la anterior y además con los pronósticos de tres expertos humanos (representados por las letras A, B y C).

Ambas bases de datos fueron preprocesadas para facilitar la aplicación automática de los diferentes modelos. En el preprocesado los datos se convierten a valores binarios, asociando el valor 1 a la presencia de una evidencia y el valor 0 a la ausencia de dicha evidencia. Los valores de los expertos humanos se codifican en cuatro valores correspondiendo con las cuatro posibles etiquetas semánticas que puede tomar el pronóstico. De la misma forma los resultados del pronóstico se codifican en dos valores. Un ejemplo de los datos con este formato puede verse en la Figura 9.12.

EVIDENCIAS								REF	EXPERTOS		
EVI1	EVI2	EVI3	EVI4	EVI5	EVI6	EVI7	EVI8	OUT	A	B	C
0	1	0	0	1	1	0	0	0	0	1	0
0	0	0	0	0	0	0	1	0	1	1	0
0	1	0	0	1	1	0	0	1	2	3	2
1	0	0	1	0	0	0	0	0	2	0	0
1	0	0	1	0	0	0	0	0	0	0	1

Figura 9.12 Ejemplo de datos preprocesados.

Medidas de pares y de grupo

Para entrenar los distintos modelos de predicción empleados se utilizó la base de datos de 3209 casos utilizando como referencia el resultado final del pronóstico y la base de 177 casos utilizando como referencia los pronósticos del experto de mayor prestigio. En el estudio comparativo se incluyeron aquellas aproximaciones que obtenían mejores resultados y que son:

ES: El modelo de factores de certidumbre actualmente implementado en el sistema y que fue entrenado con la base de datos de 3209 casos (que toman como referencia el resultado final del pronóstico).

BA1: El modelo de Bayes entrenado con los 3209 casos.

BA2: El modelo de Bayes entrenado con los 177 casos (que toman como referencia los pronósticos del experto más prestigioso).

DIS: El modelo del análisis discriminante entrenado con los 177 casos.

RNA: El modelo de neuronas artificiales entrenado con los 177 casos.

Los resultados del análisis cluster y del escalamiento multidimensional para los resultado de kappa ponderada y el porcentaje de acuerdo se pueden ver en la Figura 9.13.

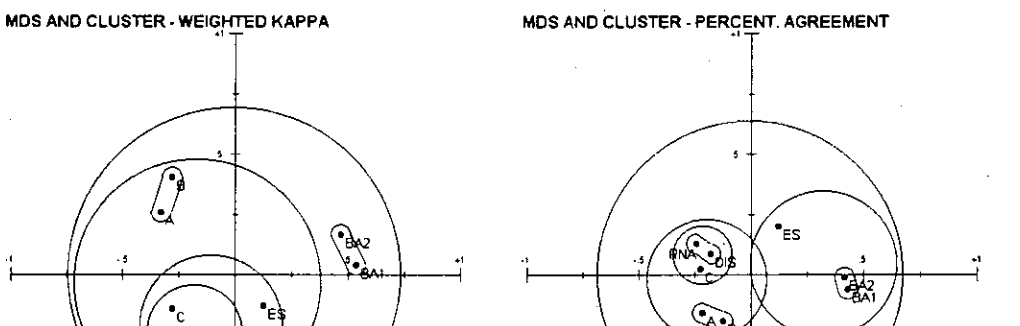


Figura 9.13 Resultados de la validación contra los expertos.

Ratios de acuerdo

Como en este sistema existe una referencia estándar que nos indica si el pronóstico realizado es correcto o no, podemos utilizar los ratios de acuerdo para ver el grado de acuerdo existente entre las distintas aproximaciones empleadas (incluyendo a los expertos humanos) y dicha referencia. Los resultados se muestran en la Tabla 9.5 en donde TP representa el ratio de verdaderos positivos (se pronostico acertadamente buena salida), TN el ratio de verdaderos negativos (se pronosticó acertadamente mala salida), FP el ratio de falsos positivos (se pronosticó erróneamente buena salida) y FN el ratio falsos negativos (se pronosticó erróneamente mala salida).

	A	B	C	ES	BA1	BA2	DIS	RNA
TP	0.767	0.781	0.863	0.548	0.255	0.269	0.822	0.863
TN	0.516	0.419	0.226	0.742	0.774	0.731	0.355	0.419
FP	0.484	0.581	0.774	0.258	0.226	0.269	0.645	0.581
FN	0.233	0.219	0.137	0.452	0.745	0.731	0.178	0.137

Tabla 9.5 Resultados de la validación contra el problema.

9.2.3. Fase de interpretación

Si atendemos a los resultados de la validación contra los expertos podemos ver que los expertos A y B presentan pronósticos muy similares. Cerca de estos expertos se sitúa el experto C, el modelo del análisis discriminante (DIS) y el de neuronas artificiales (RNA). Los resultados del sistema experto utilizando factores de certeza

(ES) parecen encontrarse a medio camino entre los resultados de los expertos humanos y los resultados de las dos aproximaciones de Bayes.

De esto se podría deducir que los modelos que presentan un mejor rendimiento son RNA y DIS, o por los menos, que son las métodos cuyo rendimiento es el más similar a el de los expertos humanos (algo que puede parecer lógico ya que han sido entrenados con datos provenientes de un experto humano).

Sin embargo, si atendemos a los resultados de la validación contra el experto nuestras interpretaciones iniciales pueden cambiar. En este tipo de validación se busca que la tasa de verdaderos positivos y verdaderos negativos sea los más elevada posible. Pero en sistemas expertos médicos muchas veces es conveniente que la tasa de falsos positivos sea poco elevada, es decir, que las veces que se diagnostique salida buena y al final sea mala sean muy pocos. Es preferible que el sistema experto sea algo "pesimista" y que clasifique un caso como malo aunque no lo sea, a que etiquete como "buenos" casos potencialmente peligrosos.

Analizando los datos de la Tabla 9.5 vemos que los expertos humanos, DIS y RNA presentan tasas de TP elevadas pero también tasas de FP elevadas (del orden del 0.7 en el caso del experto C). Por otro lado tenemos las aproximaciones de Bayes que presentan valores bajos de FP pero también valores muy bajos de TP. Sin embargo, el sistema experto utilizando CFs consigue valores de FP muy bajos junto con valores de TP altos (si bien no tanto como los expertos humanos). Esto indica que el sistema experto utilizando CFs tiene un comportamiento prudente, siendo pocos los casos potencialmente "malos" que son etiquetados como "buenos".

9.3. Resumen

En este capítulo hemos visto como la metodología de validación propuesta es aplicada a la validación de dos sistemas expertos reales. La primera fase de la metodología, la fase de planificación, nos permite determinar como llevar a cabo la validación en base a las características del dominio, del sistema y de la fase de desarrollo. Para ello podemos contar con la ayuda del sistema experto de planificación incluido en SHIVA.

La segunda fase de la metodología, la fase de aplicación, consiste en la aplicación de las medidas estadísticas según las recomendaciones establecidas en la fase de planificación. Es en esta fase cuando la ayuda de la herramienta SHIVA se hace más patente suministrando de información gráfica fácil de interpretar.

La última fase de la metodología comprende la interpretación de los resultados obtenidos en la fase anterior. Los resultados de esta fase se utilizan no sólo para comprobar que el funcionamiento del sistema es correcto, sino también para adquirir nuevo conocimiento o para refinar el conocimiento ya existente.

10. DISCUSIÓN, CONCLUSIONES, PRINCIPALES APORTACIONES Y TRABAJO FUTURO

10.1. Discusión

> La discusión estimula la curiosidad, y la curiosidad estimula la invención.
> *Charles Tomlinson, en un tratado sobre la construcción de cerraduras en 1850.*
>
> Una discusión libre y sincera es la mejor amiga de la verdad
> *G. Campbell*

En este trabajo hemos tratado de aproximarnos al complejo campo de la validación de sistemas expertos, incluyendo aspectos que abarcan la ingeniería del software, la ingeniería del conocimiento, y la estadística, en forma de procedimientos comunes a otras áreas de investigación.

La validación del software siempre ha sido una fase problemática (la única forma de asegurarnos que un programa no falla nunca es revisar todos los posibles caminos de ejecución, algo que, por supuesto, es imposible). Además, no debe verse como algo aparte de la construcción del software sino que es una fase más de su metodología de desarrollo.

Después de años de experiencia, la ingeniería del software ha desarrollado métodos y técnicas para construir programas que se basan en una división del proceso en etapas. En un principio estas etapas eran rígidas, y debían seguirse escrupulosamente paso a paso. Actualmente se puede demostrar que este enfoque sólo es adecuado en entornos deterministas y poco cambiantes. Para el resto de los entornos se sugieren técnicas que, aunque siguen el modelo en fases, lo hacen desde una perspectiva incremental.

Los sistemas expertos siguen siendo software y, por tanto, todo lo dicho para el software convencional también es aplicable a los sistemas expertos. Sin embargo, tales sistemas son muy ambiciosos, ya que pretenden ser un modelo del conocimiento del experto humano, sobre un dominio reducido pero complejo. Esto provoca que existan muchas diferencias entre los sistemas expertos y el software convencional, que se centran sobre todo en la estructura, los problemas a tratar, las estrategias de resolución, la naturaleza del conocimiento empleado y el tipo de información utilizada.

Sin embargo, la ingeniería del conocimiento no tiene por qué partir de cero, y la experiencia obtenida por años de aplicación de la ingeniería del software no debe desecharse. Así, el primer punto en el que se ve una clara influencia de la ingeniería del software en los sistemas expertos es en el desarrollo de metodologías de construcción. Generalmente los ingenieros del conocimiento se decantan por metodologías incrementales o evolutivas, porque los dominios en los que se mueven no permiten el establecimiento de fases rígidas (entornos desconocidos o cambiantes, con frecuentes realimentaciones y cambios en los requisitos iniciales). De estas metodologías incrementales la que más atención ha suscitado en los últimos tiempos es la metodología en espiral desarrollada por Boehm (1988).

Las metodologías de construcción de los sistemas expertos también incluyen una fase de prueba, en la que debe comprobarse que el sistema se ha construido correctamente y ofrece los resultados esperados. Las diversas metodologías de prueba o

de análisis del comportamiento del sistema experto tampoco tienen por qué partir de cero, y pueden basarse en las metodologías de prueba desarrolladas por la ingeniería del software. Entrando en el detalle de la estructura interna de un sistema experto, la principal preocupación del ingeniero del conocimiento es la validación de la base de conocimientos. Esto es así porque las otras partes del sistema, o bien son sistemas comerciales cuya validez se presupone (shells), o se trata de partes meramente algorítmicas cuya validación puede llevarse a cabo a partir de técnicas tradicionales (por ejemplo, el interfaz del sistema experto).

El análisis del comportamiento de un sistema experto se estructura en una pirámide cuya base son las fases de verificación y validación (conocidas popularmente como V&V). Quizá la mejor definición de estas fases es la ofrecida por Boehm (1981) por ser breves y directas:

Verificación: ¿Estamos construyendo el producto correctamente?
Validación: ¿Estamos construyendo el producto correcto?

Es decir, la verificación se encargaría de comprobar que el sistema desarrollado cumple sus especificaciones y no contiene errores y la validación se encargaría de comprobar si los resultados del sistema son correctos y cumplen las necesidades y requisitos de los usuarios.

Si el proceso de V&V es importante en los sistemas convencionales, en los sistemas expertos lo es mucho más. Esto es debido a la propia naturaleza de los sistemas expertos, que pretenden emular el comportamiento de un experto humano, aunque sea en un dominio reducido. Además, sólo aquellos sistemas validados satisfactoriamente podrán asegurar su aceptabilidad en un entorno real de trabajo (máxime si se trata de entornos críticos).

El proceso de verificación ha sido estudiado exhaustivamente, sobre todo en sistemas que utilizan las reglas de producción como método para la representación del conocimientos. Se han diseñado métodos que permiten verificar la consistencia y la completitud de una base de conocimientos, y herramientas que permiten aplicar dichos métodos dentro de shells comerciales. Las herramientas de verificación han alcanzado un alto grado de sofisticación; sin embargo todavía se les sigue achacando que, para que su funcionamiento sea efectivo, la estructura de la base de reglas tiene que ser muy sencilla.

En un reciente estudio, Murrel y Plant (1997) descubrieron que la mayoría de las herramientas de validación utilizaban técnicas comunes (análisis de causa-efecto, chequeos lógicos, comprobaciones de la estructura, etc.) y que, sin embargo, muchas técnicas de verificación encontradas en la bibliografía no habían sido implementadas por ninguna herramienta (por ejemplo, métodos formales, métodos semiformales, técnicas de análisis de fallos, etc.). Evidentemente estas técnicas son un área de futuro desarrollo en la construcción de herramientas de verificación; sin embargo, en algunas ocasiones la ausencia de estos métodos en las herramientas es debida a la falta de una investigación profunda sobre la base teórica en que se basan, o a dificultades en la aplicabilidad de dicho método.

La exposición de la fase de verificación nos sirve como introducción de la fase posterior, la validación, que es el objetivo fundamental de este trabajo. La fase de

validación se puede dividir en dos subfases con diferente objetivo. Una validación orientada a los resultados y una validación orientada al uso. Generalmente la primera suele ser un pre-requisito de la segunda (si los resultados del sistema no son aceptables no importa que su uso sea sencillo, el usuario no lo utilizará). Por ello nos centraremos en el análisis de la validación orientada a los resultados.

Dentro de este tipo de validación existen una serie de aspectos que es necesario determinar: personal involucrado en la validación (expertos, usuarios, ingenieros del conocimiento, evaluadores independientes), partes del sistema a validar (estructuras de razonamiento, resultados intermedios, resultados finales), datos utilizados en la validación (muestras aleatorias, muestras estratificadas, etc.), criterios de validación (contra el experto, contra el problema), momento en el que se realiza la validación (cuando el sistema está completo, durante el desarrollo), métodos de validación (cualitativos o cuantitativos), y errores cometidos en la validación (errores de comisión, de omisión, de riesgo para el desarrollador o de riesgo para el usuario).

Todos los aspectos vistos en la validación son importantes aunque, de entre todos, quizá debiéramos destacar los criterios de validación y los métodos de validación. Cuando validamos el software convencional no resulta especialmente complicado comprobar si un resultado es correcto o no, sin embargo, con los sistemas expertos esto no es tan trivial. No es fácil disponer de un estándar adecuado que nos indique si la solución es la correcta, generalmente podrán existir varias soluciones al mismo problema o incluso se admite un cierto nivel de incertidumbre en las soluciones.

En cuanto a los métodos de validación, existen muchas y variadas formas de validar un sistema experto. En este trabajo hemos distinguido dos grupos principales: cualitativos y cuantitativos, pero su aplicación debe ser conjunta y coordinada. Así, se puede realizar un análisis de sensibilidad a partir de medidas de exactitud, o se pueden desarrollar pruebas de Turing y luego comprobar los resultados en base a medidas de acuerdo.

Este trabajo hace especial énfasis en tres tipos de medidas estadísticas: medidas de pares, medidas de grupo y ratios de acuerdo. Las medidas de pares se utilizan para comprobar el grado de acuerdo o asociación existente entre dos expertos. Dentro de las medidas de acuerdo se han incluido el porcentaje de acuerdo, el porcentaje de acuerdo dentro de uno, kappa y kappa ponderada. El porcentaje de acuerdo es una de las medidas más populares empleadas para comprobar la fiabilidad de un sistema, aunque adolece de ciertas carencias que las otras medidas intentan solucionar. Así, el porcentaje de acuerdo incluye como acuerdos parciales aquellas discrepancias que son debidas a cuestiones de matiz, y las medidas kappa corrigen aquellos acuerdos debidos a la casualidad. Sin embargo, estas medidas tampoco están exentas de problemas, y pueden llevar a situaciones como la *saturación* del porcentaje de acuerdo dentro de uno (que consiste en que, cuando el número de categorías no es muy elevado, este porcentaje siempre suele obtener valores elevados de acuerdo), o la presencia de valores de kappa muy bajos cuando se esperaba la presencia de acuerdos altos (generalmente debidos a muestras poco balanceadas).

En los tests de pares también se han incluido medidas de asociación no paramétricas como la tau de Kendall, la gamma de Goodman-Kruskal o la rho de Spearman. Estas medidas tratan de medir el grado de asociación lineal existente entre las interpretaciones de dos expertos. De ellas, la que presenta las características más

adecuadas para ser usada en entornos de validación es la rho de Spearman, por realizar un tratamiento más racional de las observaciones ligadas.

Un aspecto importante de este trabajo es que no se detiene en la mera aplicación de las medidas de pares, sino que utiliza éstas como base para el desarrollo de las medidas de grupo. Estas medidas de grupo permiten hacernos una idea global sobre cómo se sitúan los distintos expertos según la similitud de sus diagnósticos, y además nos permiten hacerlo de una forma sencilla, gráfica y visual. Las medidas de grupo estudiadas en este trabajo son las medidas de Williams, el análisis cluster, el escalamiento multidimensional, y las medidas de dispersión y tendencia. De estas medidas las que más se utilizan son las resultantes del análisis cluster y del MDS, generalmente de forma conjunta en un único gráfico (gráfico de burbujas). Estos gráficos, novedosos en trabajos sobre la validación de sistemas expertos, permiten identificar la posición relativa de las interpretaciones de los distintos expertos con una simple observación. En base a esta información resulta más fácil realizar posteriormente un análisis detallado de los resultados de los tests de pares, la tablas de contingencia, etc.

También se analizaron los ratios de acuerdo, útiles sobre todo a la hora de comparar los resultados del sistema con los resultados de un estándar (bien sea un experto de reconocido prestigio, un consenso entre expertos o la solución real del problema). El uso de los ratios de acuerdo se emplea con frecuencia en el entrenamiento y validación de mecanismos de aprendizaje automático (como pueden ser las redes de neuronas artificiales).

El trabajo realizado con los métodos estadísticos ha sido doble: por un lado analizarlos en profundidad, intentando descubrir sus puntos fuertes y débiles a partir de diversas fuentes de la bibliografía y, por otro lado, analizar los aspectos más importantes de estas medidas cuando se utilizan en entornos de validación.

En base a las características analizadas en la validación de sistemas expertos y a las características de los distintos modelos estadísticos estudiados, se decidió desarrollar una metodología que fuera aplicable a la validación orientada a los resultados de los sistemas expertos. Esta metodología pretende paliar uno de los principales problemas que presentan los métodos de validación de sistemas expertos, como son su naturaleza *ad hoc* e informal.

La metodología propuesta se compone de tres fases fundamentales: planificación, aplicación e interpretación. La fase de planificación consiste en analizar las características del dominio de aplicación del sistema, las características del propio sistema a validar y las características de la fase de desarrollo en la que nos encontramos, para determinar cómo debería llevarse a cabo la validación (validación contra el experto, contra el problema, qué medidas utilizar, etc.)

El segundo paso de la metodología es la aplicación de las medidas propuestas en la fase de planificación y comprende las subfases de captura de la casuística, preprocesado de los datos y realización de las medidas estadísticas. Como resultado de la fase de aplicación obtenemos un conjunto de gráficos, tablas y resultados cuya interpretación se llevará a cabo en la siguiente fase de la metodología, la fase de interpretación. Esta fase es especialmente complicada ya que, medidas iguales obtenidas en contextos distintos pueden tener interpretaciones radicalmente diferentes.

Para facilitar la aplicación de la metodología propuesta se desarrolló la herramienta de validación SHIVA, un Sistema Heurístico e Integrado de Validación. Esta herramienta abarca la fase de planificación (a través de un sistema experto) y la fase de aplicación (implementando los distintos métodos estadísticos y mostrando sus resultados a través de un navegador gráfico y un navegador en modo texto).

Es importante destacar que la utilización de SHIVA facilita enormemente la aplicación de la metodología propuesta, ya que los paquetes estadísticos tradicionales no suelen incluir algunas de las medidas propuestas, o no están adaptados para realizar de forma sencilla los pasos de la metodología (por ejemplo, la utilización de los resultados de los tests de pares como información de entrada a los tests de grupo). Además, la distinta información gráfica que se puede obtener de forma sencilla mediante SHIVA (gráficos de barras, gráficos de líneas, dendrogramas, gráficos del MDS, gráficos de burbujas, etc.) facilita en gran manera la interpretación de los resultados. Al respecto podemos decir que los usuarios de la herramienta SHIVA generalmente pedían primero la información de los dendrogramas o del MDS, y una vez se habían hecho una composición de lugar, pasaban a analizar los datos de las tablas de contingencia, pares de acuerdo, etc.

En cuanto a la fase de interpretación, SHIVA implementa un sencillo sistema experto que utiliza las reglas de interpretación del índice kappa desarrolladas por Landis y Koch. Sin embargo, no se desarrolló un sistema experto completo de interpretación, en parte porque su complejidad sobrepasa los límites de la propia herramienta, y en parte porque constituye una línea en curso de investigación y desarrollo.

Por último, se evaluó la aplicabilidad de la herramienta SHIVA sobre el sistema de monitorización inteligente PATRICIA (Moret, Mosqueira y Alonso, 1997) y sobre el sistema experto antenatal NST-EXPERT (Alonso, Mosqueira y Baldonedo, 1997), ambos desarrollados en el laboratorio LIDIA. De esta forma se pudo comprobar cómo la herramienta no sólo facilita la validación de los sistemas, sino que permite el descubrimiento de nueva información y facilita el refinamiento del conocimiento ya existente.

Al respecto puede resultar interesante una anécdota ocurrida en la validación de estos sistemas: Preguntado un experto humano sobre su disponibilidad para evaluar una serie de casos para luego comparar sus interpretaciones con las del sistema experto, su primera respuesta fue: "¿queréis comprobar si yo me equivoco?", y posteriormente añadió: "estándares sobre estas interpretaciones están disponibles en la bibliografía". Esto pone de manifiesto una de las dificultades de la elección de expertos como criterio en la validación: Muchas veces los expertos humanos interpretan que a quién se está validando es a ellos y no al sistema. Respecto al segundo comentario, el conocimiento fruto de la experiencia no corresponde exactamente a un conjunto de reglas extraídas de un manual. Si así fuera haría mucho tiempo que los sistemas expertos se habrían popularizado en la industria y los estudiantes recién titulados tendrían un nivel muy superior al actual (ya que son los que mejor se conocen la información extraída de manuales). Sin embargo el conocimiento experto es algo más, son aquellos detalles que no aparecen en ningún libro pero que un experto puede reconocer fruto de sus muchos años de trabajo. Estos aspectos son los que intentan incluir los sistemas expertos en su base de conocimientos y por lo que es necesaria la presencia de un experto en el desarrollo y validación de este tipo de sistemas.

10.2. Conclusiones y principales aportaciones

> Una conclusión es el lugar donde llegaste cansado de pensar
> *Anónimo*
>
> He llegado a la conclusión de que la política es demasiado seria como para ser dejada en manos de los políticos
> *Charles DeGaulle (Militar y político francés, 1890 – 1970)*

Entre las principales conclusiones a las que hemos llegado con este trabajo podemos citar las siguientes:

- La validación es un paso crucial dentro de la metodología de desarrollo de los sistemas expertos, y garantiza su aplicabilidad en entornos reales.
- Las metodologías de desarrollo, y las metodologías de validación de los sistemas expertos, se basan en la experiencia previa extraída de la ingeniería del software.
- Es necesario tener en cuenta las características distintivas de los sistemas expertos a la hora de su construcción y de su validación.
- Los principales métodos de desarrollo de sistemas expertos se basan en metodologías evolutivas o incrementales, en las que debe incluirse una fase de V&V.
- Normalmente las validaciones de los sistemas expertos se realizan de manera informal y *ad hoc*.
- La verificación es un paso previo a la validación que nos permite comprobar si el sistema se ha desarrollado correctamente.
- La validación comprueba que el sistema realmente hace que lo que se pretende que haga.
- El desarrollo de una metodología de validación y una herramienta que soporte dicha metodología facilita enormemente la realización de la fase de validación.

Entre las principales aportaciones podemos destacar:

- El estudio de los métodos de desarrollo del software convencional y su influencia en los métodos de desarrollo de los sistemas expertos.
- La caracterización de las fases de verificación y validación, tanto de los sistemas convencionales como de los sistemas expertos
- El desarrollo de una metodología de validación que permita llevar a cabo dicha validación de una manera formal.
- El estudio de diversos métodos estadísticos y su aplicación al campo de la validación. Dentro de estos métodos estadísticos se puede resaltar cómo la combinación de técnicas, que normalmente se utilizan por separado, permite

la obtención de mejores resultados que facilitan la interpretación de la validación.

- La construcción de una herramienta (SHIVA) que permite la fácil aplicación de las distintas fases de la metodología de validación.

- La utilización de SHIVA sobre sistemas reales para demostrar su aplicabilidad.

10.3. Trabajo futuro

> Nunca pienso en el futuro. Llega enseguida
> *Albert Einstein (Físico y matématico de origen alemán. 1.879 - 1.955)*

Como líneas de trabajo futuro podemos incluir las siguientes:

- Incluir en el sistema soporte para la validación orientada al uso (implementando, por ejemplo, el método AHP).

- Estudiar e implementar los métodos matemáticos para la construcción de estándares.

- Estudiar e implementar nuevas medidas de acuerdo (medidas de Fleiss, medidas de Light, la C de Ciccheti, etc.) y medidas de asociación predictiva (lambda y tau de Goodman-Kruskal, eta de Theil, etc.).

- Ultimar la implementación del sistema experto de interpretación.

- Validar nuevos sistemas expertos, como el sistema de monitorización inteligente de apneas en sueño (MIDAS) que está siendo desarrollado en estos momentos en el Laboratorio LIDIA de la Universidade de A Coruña.

Como resumen final podemos decir que el campo de la verificación y la validación de los sistemas expertos es crucial en el desarrollo de los mismos. Si somos capaces de validar correctamente estos sistemas, y descubrir los errores antes de que se produzcan, habremos dado un paso adelante en la aceptación de los sistemas expertos en los distintos ámbitos de trabajo.

REFERENCIAS

Por necesidad, por afición o por placer, todos citamos a los demás
Ralph Waldo Emerson (Poeta y pensador estadounidense. 1.803 – 1.882).

Si quieres ideas nuevas, lee libros viejos.
Ivan Pavlov (Fisiólogo y premio nobel ruso. 1849 – 1936).

Los libros son, entre mis consejeros, los que más me agradan, porque ni el temor ni la ambición les impiden decirme lo que debo hacer.
Alfonso II

(Adelman, 1991a) Adelman, L. *Evaluating Decision Support and Expert Systems*, Wiley, New York, 1991.

(Adelman, 1991b) Adelman, L. "Experiments, Quasi-Experiments, and Case Studies: A Review of Empirical Methods for Evaluating Decision Support Systems." *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, no. 2, pp. 293-301, 1991.

(Adlassnig y Scheithauer, 1989) Adlassnig, K.P. and Scheithauer, W. "Performance Evaluation of Medical Expert Systems Using ROC Curves." *Computers and Biomedical Research*, vol. 22, pp. 297-313, 1989.

(Adrion et al., 1982) Adrion, W.R., Branstad, M.A., Cherniavsky, J. "Validation, verification and testing of computer software." *Computing Surveys*, vol. 14, no. 2, pp. 159-192, 1982.

(Agarwal y Tanniru, 1992) Agarwal, R. and Tanniru, M. "A Petri-net Approach for Verifying the Integrity of Production Systems." *International Journal of Man-Machine Studies*, vol. 26, pp. 447-468, 1992.

(Aho, Sethi y Ullman, 1990) Aho, A.V., Sethi, R. and Ullman, J.D. *Compiladores: Principios, Técnicas y Herramientas*. Addison-Wesley iberoamericana, 1990.

(Aldenderfer y Blashfield, 1984) Aldenderfer, M.S. and Blashfield, R.K. *Cluster Analysis*, Sage University Paper series on Quantitative Applications in the Social Sciences, no. 44, Beverly Hills and London, Sage Pubns., 1984.

(Alonso et al., 1992) Alonso-Betanzos, A., Moret-Bonillo, V., Devoe, L.D., Searle, J.R., Banias, B. and Ramos, E. "Computerized Antenatal Assessment: The 'NST-EXPERT' Project." *Automedica*, vol. 14, pp. 3-22, 1992.

(Alonso et al., 1995) Alonso-Betanzos, A., Guijarro-Berdiñas, B., Moret-Bonillo, V., and López-González, S. "The NST-EXPERT project: the need to evolve." *Artificial Intelligence in Medicine*, vol. 7, pp. 297-313, 1995.

(Alonso, Mosqueira y Baldonedo, 1997) Alonso-Betanzos, A., Mosqueira-Rey, E. and Baldonedo del Río, B. "A Comparative Analysis of the Neonatal Prognosis Problem Using Artificial Neural Networks, Statistical Techniques and Certainty Management Techniques." in *Biological and Artificial Computation: From Neuroscience to Technology*, Lecture Notes in Computer Science, vol. 1240, Springer-Verlag, pp. 995-1004, 1997.

(Amescua et al., 1995) de Amescua Seco A., García Sánchez, L., Martínez Fernández, P., Díaz Pérez, P. *Ingenieria del Software de Gestión: Análisis y diseño de aplicaciónes*. Ed. Paraninfo, Madrid, 1995.

(Anderberg, 1973) Anderberg, M.R. *Cluster Analysis for Applications*, Academic Press, New York, 1973.

(Anderson, 1960) Anderson, E. "Some stochastic process models for intelligence test scores." in *Mathematical Methods in the Social Sciences*, K.J. Arrow et al. (eds.). Stanford University Press, Stanford, CA, pp. 205-220, 1960.

(Andrews, 1960) Andrews, D.F. "Plots of high-dimensional data." *Biometrics*, vol. 28, pp. 125-136, 1972.

(Arce, 1993) Arce, C. *Escalamiento Multidimensional*, PPU, Barcelona, 1993.

(Bachant y McDermott, 1984)* Bachant, J. and McDermott, J. "R1 Revisited: Four Years in the Trenches." *AI Magazine*, vol. 5, no. 3, pp. 21-32, 1984.

(Bahill et al., 1995) Bahill, A.T., Bharathan, K. and Curlee, R.F. "How the testing techniques for a decision support system changed over nine years." *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, no. 12, pp. 1533-1542, December, 1995.

(Barthélemy et al., 1987) Barthélemy, S., Edin, G., Toutain, E. and Becker, S. *Requirements analysis in KBS development.* ESPRIT Project P1098 Deliverable D3 (task A2), Cap Sogeti Innovation.

(Bauer, 1972) Bauer, F.L. "Software Engineering." *Information Processing*, vol. 71, North Holland Publishing Co., Amsterdam, 1972.

(Berliner, 1980) Berliner, H. "Backgammon Program Beats World Champ." *SIGART Newsletter*, no. 69, January, pp. 6-9, 1980.

(Bisquerra, 1989) Bisquerra R. *Introducción conceptual al análisis multivariable: un enfoque informático con los paquetes SPSS-X, BMDP, LISREL y SPAD. Vol. II.* Ediciones PPU, Barcelon, 1989.

(Boehm, 1976) Boehm, B.W. "Software Engineering." *IEEE Transactions on Computers*, C-25, no. 2, pp. 1261-1241, Dec 1976.

(Boehm, 1981) Boehm, B.W. *Software Engineering Economics.* Prentice-Hall, Inc. Englewood Cliffs, NJ, 1981.

(Boehm, 1983) Boehm, B.W. "Seven Basic Principles of Software Engineering." *Journal of Systems and Software*, no. 3, pp. 3-24, 1983.

(Boehm, 1988)* Boehm, B.W. "A Spiral Model of Software Development and Enhancement." *Computer*, May, pp. 61-72, 1988.

(Bonner, 1964) Bonner, R.E. "On some clustering techniques.", *I.B.M.J. Res Dev.*, vol. 8, pp. 22-32, 1964.

(Boose, 1986) Boose, J.H., *Expertise Transfer for Expert System Design.* Elsevier, New York, 1986.

(Boose y Bradshaw, 1987) Boose, J.H., and Bradshaw, J. "Expertise Transfer and Complex Problems Using Aquinas as a Knowledge Acquisition Workbench for Expert Systems." *International Journal of Man-Machine Studies*, vol. 26, pp. 3-28, 1987.

(Borrajo et al., 1993) Borrajo, D., Juristo, N., Martínez, V. y Pazos, J. *Inteligencia Artificial: Métodos y Técnicas*, Ed. Centro de Estudios Ramón Areces, S.A., Madrid, 1993.

(Botting, 1985) Botting, P.J. "Prototypes vs. Mock-ups vs. Breadboards." *ACM SIGSOFT Software Engineering Notes*, vol. 10, no. 1, p. 18, January 1985.

(Buchanan et al., 1983) Buchanan, B.G., Barstow, D., Bechtal, R., Bennett, J., Clancey, W., Kulikowski, C. Mitchell, T., and Waterman, D.A. "Constructing an Expert System." in *Building Expert Systems*, F. Hayes-Roth, D.A. Waterman, D.B. Lenat (eds.), Addison-Wesley Pub. Co., Reading, MA, 1983.

(Buchanan y Shortliffe, 1984) Buchanan, B.G. and Shortliffe, E.H. *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Project*, Addison-Wesley, Reading, MA.

(Cailliez, 1983) Cailliez, F. "The Analytical Solution of the Additive Constant Problem." *Psychometrika*, vol. 48, pp. 305-308, 1983.

(Cardeñosa et al., 1991) Cardeñosa, J., Juristo, N., Morant, J.L., and Pazos, J. "The Knowledge Engineer in the Knowledge Industry." *The Journal of Knowledge Engineering*, vol. 4, no. 2, pp. 43-53, Summer 1991.

(Carrico et al., 1989) Carrico, M.A., Girard J.E., and Jones J.P. *Building Knowledge Systems.* McGraw-Hill Book Company, New York, 1989.

(Carroll y Arabie, 1980) Carroll, J.D. and Arabie, P. "Multidimensional Scaling." in *Annual Review of Psychology*, M.R. Rosenweig y L.W. Porter (eds.), vol. 31, pp. 607-649, Annual Reviews, Palo Alto, CA, 1980.

(Carroll y Chang, 1970) Carroll, J.D. and Chang J.J. "Analysis of individual differences in multidimensional scaling via *N*-way generalization of Eckart-Young decomposition." *Psychometrika*, vol. 35, pp. 283-319, 1970.

(Castillo y Alvarez, 1989) Castillo, E. y Alvarez, E. *Sistemas Expertos: Aprendizaje e Incertidumbre*, Ed. Paraninfo, Madrid, 1989.

(Chandrasekaran, 1983)* Chandrasekaran, B. "On Evaluating AI Systems for Medical Diagnosis." *AI Magazine*, vol. 4, no. 2, pp. 34-37, 1983.

(Cheng, 1989) Cheng, A.M. "Expert System Validation as it Applies to Expert Systems Utilizing a Frame-Based Knowledge Representation." Master's Thesis, Department of Computer Science and Engineering, University of South Florida, Tampa, FL, 1989.

(Chernoff, 1973) Chernoff, H. "Using faces to represent points in a k-dimensional space graphically." *Journal of the American Statistical Association*, vol. 68, pp. 361-368, 1973.

(Clarke, 1977) Clarke, A.C. "Quarantine." *Isacc Asimov's Science Fiction Magazine*, First Issue, vol. 1, no. 1, Spring 1977, also on-line in URL: http://www.chess.ibm.com/learn/html/e.8.2.html

(Clarke et al., 1994) Clarke, K., O'Moore, R., Smeets, R., Talmon, J., Brender, J., McNair, P., Nykanen, P., Grimson, J. and Barber, B. "A methodology for evaluation of knowledge-based systems in medicine." *Artificial Intelligence in Medicine*, vol. 6, pp. 107-121, 1994.

(Cohen, 1960) Cohen, J. "A coefficient of agreement for nominal scales." *Educational and Psychological Measurement*, vol. 20, pp. 37-46, 1960.

(Cohen, 1968) Cohen, J. "Weighted kappa: nominal scale agreement with provision for scaled disagreement or partial credit." *Psychological Bulletin*, vol. 70, no. 4, pp. 213-220, October, 1968.

(Conde y Winter, 1990) Conde Lázaro, C. y Winter Althaus, G. *Métodos y algoritmos básicos del álgebra numérica*. Ed. Reverté S.A., Barcelona, 1990.

(Cormack, 1971) Cormack, R.M. "A review of classification.", *Journal of the Royal Statistical Society, Series A*, vol. 134, pp. 321-367, 1971.

(Cortés et al., 1993) Cortés, U., Béjar, J. y Moreno, A. *Inteligencia Artificial*, Edicions UPC, Barcelona, 1993.

(Cox y Cox, 1994) Cox, T.F. and Cox, M.A. *Multidimensional Scaling*, Chapman & Hall Monographs on Statistics and Applied Probability, 1994.

(Cragun y Steudel, 1987)* Cragun, J.B., Steudel, H.J. "A Decision-Table-Based Processor for Checking Completeness and Consistency in Rule-Based Expert Systems." *International Journal of Man-Machine Studies*, vol. 26, no.5, pp. 633-648, 1987.

(Cuadras, 1991) Cuadras, C.M. *Métodos de Análisis Multivariante*. PPU (Promociones y Publicaciones Universitarias, S.A.), Barcelona, 1991.

(Culbert et al., 1987)* Culbert, C., Riley, G., Savely, R.T. "Approaches to the Verification of Rule-Based Expert Systems" *SOAR'87: First Annual Workshop on Space Operation Automation and Robotics*, pp. 27-37, 1987.

(Daniels, 1950) Daniels, H.E. "Rank correlation and population models." *Journal of the Royal Statistical Society, B*, vol. 12, pp. 171-181, 1950.

(Davis, 1976) Davis, R. "Applications of meta-level knowledge to the construction, maintenance and use of large knowledge bases." *Ph. D. Diss.* Rept. STAN-CS-76-564, Computer Science Department, Stanford University, Stanford, Calif. Reprinted in *Knowledge-based systems in artificial intelligence*, R. Davis and D.B. Lenat (eds.). New York, McGraw-Hill, 1980.

(Dempster, 1967) Dempster, A. "Upper and lower probabilities induced by a multivalued mapping." *Annals of Mathematical Statistics*, vol. 38, no. 2, pp. 325-399, 1967.

(Detrano et al., 1992) Detrano, R. Bobbio, M., Olson, H. Shandling, A., Ellestad, M.H. et al. "Computer Probability Estimates of Angiographic Coronary Artery Disease: Transportability and Comparison with Cardiologists' Estimates." *Computers and Biomedical Research*, vol. 25, pp. 468-485, 1992.

(Deuchler, 1914) Deuchler, G. *Zeit. Pädagog. Psychol. Exper. Pädagog.*, vol. 15, pp. 114-131, 145-159, 229-242, 1914.

(Dillon y Goldstein, 1984) Dillon, W.R. and Goldstein, M. *Multivariate Analysis: Methods and Applications*, John Wiley and Sons, New York, 1984.

(Donker et al., 1992) Donker, D.K., Hasman, A. and Van Geijin, H.P. "Kappa Statistics: What Does It Say?." *MEDINFO'92*, K.C. Lun et al. (eds.), Elsevier Science Publishers B.V. (North-Holland), 1992.

(Dubes, 1993) Dubes, R.C. "Cluster analysis and related issues.", in *Handbook of Pattern Recognition and Computer Vision*, C.H. Chen, L.F. Pau and P.S.P. Wang (eds.), World Scientific Publishing Company, pp. 3-32, 1993.

(Enc. Stat. Sci., 1981) *Encyclopedia of Statistical Sciences*, Kotz, S. and Johnson, N.L. (eds.), John Wiley and Sons, New York, 1981.

(Esscher, 1924) Esscher, F. *Skand. Aktuarietidskr*, vol. 7, pp. 201-219, 1924.

(Everitt, 1993) Everitt, B.S. *Cluster Analysis*, Edward Arnold, London, 1993.

(Famili et al. 1996) Famili, A., Shen W.M., Weber, R. and Simoudis, E. "Data Preprocessing and Intelligent Data Analysis." *Intelligent Data Analysis*, vol. 1, no. 1, 1996.

(Fechner, 1897) Fechner, G.T. *Kollektivmasslehre*. W.Engelmann, Leipzig, 1897.

(Feigenbaum, 1979) Feigenbaum, E.A., "Themes and Case Studies of Knowledge Engineering." in *Expert Systems in the Micro-Electronic Age*, D. Michie (ed.), Edinburgh University Press, Edimburgh, Scotland, pp. 3-25, 1979.

(Fleiss y Zubin, 1969) Fleiss, J.L. and Zubin, J. "On the methods and theory of clustering." *Multivariate Behaviour Res.*, vol. 4, pp. 235-250, 1969.

(Fleiss et al., 1969) Fleiss, J.L., Cohen, J., and Everitt, B.S. "Large sample standard errors of kappa and weighted kappa." *Psychological Bulletin*, vol. 72, pp. 323-327, 1969.

(Fleiss, 1971) Fleiss, J.L. "Measuring nominal scale agreement among many raters." *Psychological Bulletin*, vol. 76, no. 5, pp. 378-382.

(Fleiss, 1981) Fleiss, J.L. *Statistical Methods for Rates and Proportions*, 2nd edition, John Wiley and Sons, New York, 1981.

(Gaines, 1987) Gaines, B.R. "An Overview of Knowledge Acquisition and Transfer." *International Journal of Man-Machine Studies*, vol. 26, pp. 453-472, 1987.

(Gaschnig et al. 1983) Gaschnig, J., Klahr, P., Pople, H., Shortliffe, E., and Terry, A. "Evaluation of Expert Systems: Issues and Case Studies" in *Building Expert Systems*, F. Hayes-Roth, D.A. Waterman, D.B. Lenat (eds.), Addison-Wesley Pub. Co., Reading, MA, 1983.

(Geissman y Schultz, 1988)* Geissman, J.R., Schultz, R.D., "Verification and Validation of Expert Systems." *AI Expert*, pp. 26-33, Feb 1988.

(Georgakis et al., 1990) Georgakis, D.C., Trace, D.A., Naeymi-Rad, F. and Evens, M. "A Statistical Evaluation of the Diagnostic Performance of MEDAS – the Medical Emergency Decision Assistance System." *SCAMC*, pp. 815-819, 1990.

(Gibbons, 1993) Gibbons J.D. *Nonparametric Measures of Association*, Sage University Paper series on Quantitative Applications in the Social Sciences, no. 07-091, Sage Publications, Newbury Park, California, 1993.

(Gibbons y Chakraborti, 1992) Gibbons, J.D. and Chakraborti, S. *Nonparametric statistical inference*, 3rd ed., Marcel Dekker, New York, 1992.

(Ginsberg y Weiss, 1985) Ginsberg, A. and Weiss S. "SEEK2: a generalized approach to automatic knowledge base refinement." *Proc. IJCAI*, pp. 367-374, 1985.

(Ginsberg, 1988)* Ginsberg, A. "Knowledge-Base Reduction: A New Approach to Checking Knowledge Bases for Inconsistency and Redundancy." *7th National Conference on AI*, August, 1988.

(Gjørup, 1988) Gjørup, T. "The Kappa Coefficient and the Prevalence of a Diagnosis." *Methods of Information in Medicine*, vol. 27, pp. 184-186, 1988

(Gonzalez y Dankel, 1993) Gonzalez, A.J., and Dankel, D.D. *The Engineering of Knowledge-Based Systems: Theory and Practice*. Prentice-Hall International Inc, Englewood Cliffs, New Jersey, 1993.

(Goodman y Kruskal, 1954) Goodman, L.A. and Kruskal, W.H. "Measures of association for cross classifications." *Journal of the American Statistical Association*, vol. 49, pp. 732-764, 1954 (Correction, *Ibid.*, vol. 52, pp. 578).

(Gordon, 1980) Gordon, A.D. *Classification*, Chapman & Hall, London, 1980.

(Gower, 1967) Gower, J.C. "A comparison of some methods of cluster analysis". *Biometrics*, vol. 23, pp. 623-637, 1967.

(Gower, 1971) Gower, J.C. "A general coefficient of similarity and some of its properties." *Biometrics*, vol. 27, pp. 857-872, 1971.

(Guilford, 1950) Guilford, J.P. *Fundamental Statistics in Psychology and Education, 2nd ed.* McGraw-Hill, New York, 1950.

(Gupta, 1993) Gupta U.G. "Validation and Verification of Knowledge-Based Systems: A Survey." *Journal of Applied Intelligence*, vol. 3, pp. 343-363, 1993.

(Guttman, 1941) Guttman, L. in *Prediction of Personal Adjustement* (Bull. 48), P. Horst et al. (eds.), Social Science Research Council, New York, pp. 253-318, 1941.

(Hamilton y Breslawski, 1996) Hamilton, D.M. and Breslawski, S. "Knowledge Acquisition for Multiple Site, Related Domain Expert Systems: Delphi Process and Application." *Expert Systems with Applications*, vol. 11, no. 3, pp. 377-389, 1996.

(Harmon y King, 1985) Harmon, P., King, D. *Expert Systems: Artificial Intelligence in Business*. John Wiley and Sons, Inc., 1985.

(Hickam et al., 1985) Hickam, D.H. et al. "The Treatment Advice of a Computer-Based Cancer Chemotherapy Protocol Advisor." *Annals of Internal Medicine*, vol. 103, no. 6 (Part 1), pp. 928-936, 1985.

(Hernandez et al., 1994) Hernandez, C., Sancho, J.J., Belmonte, M.A., Sierra, C. and Sanz, F. "Validation of the Medical Expert System RENOIR." *Computers and Biomedical Research*, vol. 27, pp. 456-471, 1994.

(Hoppe y Meseguer, 1993) Hoppe, T., Meseguer, P. "VVT Terminology: A Proposal." *IEEE Expert*, June, pp. 48-55, 1993.

(Hsu, 1990) Hsu, F., Anantharaman, T., Campbell, M., and Nowatzyk, A. "A Grandmaster Chess Machine." *Scientific American*, vol. 263, no. 4, October, pp. 44-50, 1990.

(Huff y Black, 1978) Huff, D.L. and Black, W. "A multivariate graphical display for regional analysis." in *Graphical Representation of Multivariate Data*, Peter C. C. Wang (ed.). Academic Press, New York, pp. 199-218, 1978.

(IBM, 1997) IBM Corp. (19-Dec-97) "Kasparov vs. Deep Blue: the rematch", [On-Line]. URL: http://www.chess.ibm.com/

(Jardine y Sibson, 1968) Jardine, N. and Sibson, R. "The construction of hierarchic and non-hierarchic classifications." *Comp. J.*, vol. 11, pp. 117-184, 1968.

(Jardine y Sibson, 1971) Jardine, N. and Sibson, R. *Mathematical Taxonomy*, John Wiley and Sons, New York.

(Jobson, 1992) Jobson, J.D. *Applied Multivariate Data Analysis*, Springer-Verlag, New York, 1992.

(Jones, 1990) Jones, G.W. *Software Engineering*, John Wiley & Sons, New York, 1990.

(Kandelin y O'Leary, 1995) Kandelin, N. A., O'Leary, D.E. "Verification of Object-Oriented Systems: Domain-Dependent and Domain-Independent Approaches." *J. Systems Software*, vol. 29, no. 3, pp. 261-269, June, 1995.

(Kang y Bahill, 1990)* Kang, Y. Bahill, T. "A Tool for Detecting Expert System Errors." *AI Expert*, pp. 42-51, February, 1990.

(Kaufman y Rousseeuw, 1990) Kaufman, L. and Rousseeuw, P.J. *Finding Groups in Data*, John Wiley & Sons, New York, 1990.

(Kendall, 1938) Kendall, M.G. *Biometrika*, vol. 30, pp. 81-93, 1938.

(Kendall y Gibbons, 1990) Kendall, M.G. and Gibbons, J.D. *Rank Correlation Methods*. Edward Arnold, London, 1990.

(Kruskal, 1964a) Kruskal, J.B. "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis." *Psychometrika*, vol. 29, pp. 1-27, 1964.

(Kruskal, 1964b) Kruskal, J.B. "Nonmetric multidimensional scaling: A numerical method." *Psychometrika*, vol. 29, pp. 115-129, 1964.

(Kulczynski, 1928) Kulczynski, S. *Bull. Int. Acad. Pol. Sci. B*, vol. 2, 1928.

(Kulikowski y Weiss, 1982) Kulikowski, C.A. and Weiss, S.M. "Representation of Expert Knowledge for Consultation: the Casnet and Expert Projects." in *Artificial Intelligence in Medicine*, P. Szolovits (Ed.), Westview Press, Boulder, CO, pp. 21-56, 1982.

(Lamberti y Newsome, 1989) Lamberti, D.M., and Newsome, S.L. "Presenting abstract versus concrete information in expert systems: What is the impact on user performance?." *International Journal of Man-Machine Studies*, vol. 31, no. 7, pp. 27-45, 1989.

(Lance y Williams, 1967) Lance, G.N. and Williams, W.T. "A general theory of classificatory sorting strategies: 1. Hierarchical systems." *Comp. J.*, vol. 9, pp. 373-380, 1967.

(Landis y Koch, 1977) Landis, J.R. and Koch G.G. "The measurement of observer agreement for categorical data." *Biometrics*, vol. 33, pp. 159-174, 1977.

(Lee y O'Keefe, 1994) Lee, S. and O'Keefe R.M. "Developing a Strategy for Expert System Verification and Validation." *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 4, pp. 643-655, April 1994.

(Lethan y Jacobsen, 1987) Lethan, H., and Jacobsen, H. "ESKORT – An Expert System for Auditing VAT Accounts." *Proceedings of Expert Systems and their Applications*, Avignon, France, 1987.

(Levy, 1986) Levy, D. *Manual de ajedrez por computadora*. Editorial Mitre, Barcelona, 1986.

(Liebowitz, 1986)* Liebowitz, J. "Useful Approach for Evaluating Expert Systems." *Expert Systems*, vol. 3, no. 2, pp. 86-96, 1986.

(Light, 1971) Light, R.J. "Measures of response agreement for qualitative data: some generalizations and alternatives." *Psychological Bulletin*, vol. 76, pp. 365-377, 1971.

(Lindeberg, 1925) Lindeberg, J.W. *VI Skand. Mathematikerkongr.*, Copenhagen, pp. 437-446, 1925.

(Lindeberg, 1929) Lindeberg, J.W. *Nord. Statist. J.*, vol. 1, pp. 137-141, 1929.

(Lipps, 1906) Lipps, G.F. *Die Psychischen Massmethoden*. F. Vieweg und Sohn, Braunschweig, Germany, 1906.

(Liu y Dillon, 1991) Liu, N.K. and Dillon, T. "An Approach Towards the Verification of Expert Systems Using Numerical Petri Nets." *International Journal of Intelligent Systems*, vol. 6, pp. 255-276, 1991.

(López et al., 1990) López, B., Meseguer, P., and Plaza, E. "Knowledge Based Systems Validation: A State of the Art", *AICOM*, vol. 3, no. 2, June, 1990.

(Lowry y Duran, 1989) Lowry, M. Duran, R. "Knowledge-based Software Engineering." in *The Handbook of Artificial Intelligence*, A. Barr, P.R. Cohen, and E.A. Feigenbaum (Eds.), Addison-Wesley Pub. Co., 1989.

(Lozano y Larrañaga, 1998) Lozano, J.A. y Larrañaga, P. "Aplicación de los algoritmos genéticos al problema del clustering jerárquico." *Inteligencia Artificialk*, nº 5, primavera, 1998.

(Luger y Stubblefield, 1993) Luger, G.F., Stubblefield, W.A. *Artificial Intelligence: structures and strategies for complex problem solving*. The Benjamin/Cummings Publishing Company, Inc. Redwood City, California, USA, 1993.)

(Macleish, 1986) Macleish, K. J. "An expert system development life cycle model and its relevance to traditional software systems." *Proceedings of The International Phoenix Conference on Computers and Communications*, pp. 592-596, 1986.

(Macro y Burton, 1987) Macro, A. and Burton, J. *The Craft of Software engineering*, Addison-Wesley, Reading, 1987.

(Martin, 1987) Martin, N. "Software engineering issues in expert system development." *SoftPert Systems, Ltd.*, 1987.

(Maté y Pazos, 1988) Maté Hernández, J.L. y Pazos Sierra, J. *Ingeniería del Conocimiento: diseño y construcción de sistemas expertos*, Editorial SEPA, Córdoba (Argentina), 1988.

(Mayrhauser, 1990) von Mayrhauser, A. *Software Engineering: Methods and Management*, Academic Press, Inc. San Diego, CA, 1990.

(McConnell, 1997) McConnell, S. *Desarrollo y Gestión de Proyectos Informáticos*, McGraw-Hill (Microsoft Press), 1997.

(McCracken y Jackson, 1982) McCracken, D.D., and Jackson, M.A. "Life-Cycle Concept Considered Harmful." *ACM Software Engineering Notes*, April, pp. 29-32, 1982.

(McGraw y Harbison-Briggs, 1989) McGraw, K.L. and Harbison-Briggs, K. *Knowledge Acquisition: Principles and Guidelines*, Prentice-Hall International Editions, Englewood Cliffs, NJ, 1989.

(McLachlan, 1992) McLachlan, G.J. "Cluster analysis and related techniques in medical research.", *Statistical Methods in Medical Research*, vol. 1, pp. 27-48, 1992.

(Medsker et al., 1994) Medsker, L., Tan, M. and Turban, E. "Knowledge Acquisition from Multiple Experts: Problems and Issues." in *Moving towards expert systems globally in the 21st century*, J. Liebowitz (ed.), New York Cognizant Communication Corporation, New York, pp. 199-208, 1994.

(Michie, 1982) Michie, D. "The state of art in machine learning." in *Introductory Readings in Expert Systems*, D. Michie (ed.), pp. 208-228, 1982.

(Milligan, 1980) Milligan, G.W. "An examination of the effect of six types of error perturbation on fifteen clustering algorithms." *Psychometrika*, vol. 50, pp. 159-179, 1980.

(Milligan y Cooper, 1988) Milligan G.W. and Cooper M.C. "A study of standarizacion of variables in cluster analysis." *J. Classification*, vol. 5, pp. 181-204, 1988.

(Moret et al., 1993) Moret-Bonillo, V., Alonso-Betanzos, A., García, E., Cabrero, M., Guijarro, B. "The PATRICIA project: a semantic-based methodology for intelligent monitoring in the ICU.", *IEEE Eng. Med. Biol. Magazine*, vol. 14, pp. 59-68, 1993.

(Moret, Mosqueira y Alonso, 1997) Moret-Bonillo, V., Mosqueira-Rey, E. and Alonso-Betanzos, A. "Information Analysis and Validation of Intelligent Monitoring Systems in Intensive Care Units." *IEEE Transactions on Information Technology in Biomedicine*, vol. 1, no. 2, pp. 87-99, June, 1997.

(Morris, 1985) Morris, J. "Software Engineering an AI." *ACM SIGART Newsletter*, vol. 92, pp. 2, 1985.

(Murrel y Plant, 1997) Murrel, S. and Plant R.T. "A survey of tools for the validation and verification of knowledge-based systems: 1985- 1995." *Decisión Support Systems*, vol. 21, pp. 307-323, 1997.

(Myers, 1979) Myers, G. *The Art of Software Testing*, Wiley, 1979.

(Newell y Simon, 1961) Newell, A., and Simon, H. A. "GPS a program that simulates human thought." in *Lernende Automaten*, H. Billing (Ed.), pp. 109-129, Oldenboung, Münich, Germany.

(Nguyen et al., 1987)* Nguyen, T.A., Walton, A.P., Laffey, T.J., Pecora, D. "Knowledge Base Verification." *AI Magazine*, vol. 8, no. 2, pp. 69-75, Summer 1987.

(Noblett y Jones, 1991) Noblett, L.M. and Jones C.M., "Knowledge-Based System Development: The Need for a Methodology." *The Journal of Knowledge Engineering*, vol.4, no. 4, Winter, 1991.

(Norusis, 1995) Norusis M.J. (1995). *SPSS Professional Statistics 6.1*. SPSS Inc.

(O'Keefe et al., 1987)* O'Keefe, R.M., Balci, O., Smith, E.P. "Validating Expert System Performance." *IEEE Expert*, vol. 2, no. 4, pp. 81-89, Winter 1987.

(O'Keefe, 1989) O'Keefe, R.M. "The evaluation of decision-aiding systems: guidelines and methods." *Inform. Management*, vol. 17, pp. 217-226, 1989.

(O'Keefe y O'Leary, 1993) O'Keefe, R.M., O'Leary, D.E., "Expert system verification and validation: a survey and tutorial." *Artificial Intelligence Review*, vol. 7, no. 1, pp 3-42, 1993.

(O'Leary, 1987) O'Leary, D.E. "Validation of Expert Systems." *Decision Sciences*, vol. 18, no. 3, pp. 468-486, 1987.

(O'Leary, 1990) O'Leary, D.E. "Soliciting Weights or Probabilities from Experts for Rule-Based Systems." *International Journal of Man-Machine Studies*, vol. 32, pp. 293-301, 1990.

(O'Leary, 1993) O'Leary, D.E. "Verifying and Validating Expert Systems: A Survey.", in *Expert Systems in Business and Finance: Issues and Applications*, P.R. Watkins and L.B. Eliot (Eds.), John Wiley & Sons Ltd, pp. 181-208, 1993.

(Pearson, 1904) Pearson, K. *Draper's Co. Res. Mem. Biom. Ser.*, vol. 1, pp. 1-35, 1904.

(Pipard, 1988) Pipard, E. "Detection d'incoherences et d'incompletitudes dans les bases de regles: le systeme INDE." *Proc. AVIGNON*, pp. 15-33, 1988.

(Pirat, 1991) Pirat, J. "El nacimiento de la inteligencia artificial." *Mundo Científico*, vol. 5, no. 5, 1991.

(Platts, 1997) Platts, J., (24-Nov-97) "Notes on Knowledge Engineering", [On-Line]. URL: http://www.mdx.ac.uk/www/ai/samples/

(Poe, 1983) Poe, E. A. *Obras Selectas de Edgar Allan Poe*. Ediciones Orbis, S.A., Barcelona, 1983.

(Politakis, 1985) Politakis, P. "Empirical Analysis of Expert Systems." *Research Notes in Artificial Intelligence*, vol. 6, Pitman Publ., 1985.

(Popping, 1981) Popping, R. "Nominal Scale Agreement." in *Encyclopedia of Statistical Sciences*, Kotz, S. and Johnson, N.L. (eds.), John Wiley and Sons, New York, 1981.

(Preece et al., 1992) Preece, A.D., Shinghal, R., Batarekh, A. "Verifying Expert Systems: A Logical Framework and a Practical Tool." *Expert Systems with Applications*, vol. 5, pp. 421-436, 1992.

(Press et al., 1992) Press, W.H., Teukolsky, S.A. Vetterling, W.T. and Flannery, B.P. *Numerical Recipes in C: The Art of Scientific Computing*. 2nd Edition. Cambridge University Press, 1992.

(Pressman, 1988) Pressman, R.S. *Ingeniería del Software: Un enfoque práctico*, 2ª edición. McGraw-Hill Interamericana, Madrid, 1988.

(Pressman, 1998) Pressman, R.S. *Ingeniería del Software: Un enfoque práctico*, 4ª edición. McGraw-Hill Interamericana, Madrid, 1998.

(Reggia, 1985) Reggia, J.A. "Evaluation of medical expert systems: case study in performance assessment." *Proceedings of the 9th Annual Symposium on Computer Applications in Medical Care*, Washington, D.C., November, pp. 287-291, 1985. Reprinted in *Selected Topics in Medical Artificial Intelligence*, P.L. Miller (ed.), Springer-Verlag, New York, 1988.

(Richardson, 1938) Richardson, M.W. "Multidimensional Psychophysics." *Psychological Bulletin*, vol. 35, pp. 659-660, 1938.

(Riedel y Pitz, 1986) Riedel, S.L. and Pitz, G.F. "Utilization-oriented evaluation of decision support systems." *IEEE Trans. Sys. Man. Cybern.*, vol. SMC-16, pp. 980-996, 1986.

(Roth y Wood, 1993) Roth, R.M. and Wood, W.C. "Knowledge Acquisition from Single Versus Multiple Experts: A Field Study Comparison Using the Delphi Technique." *The Journal of Knowledge Engineering*, vol. 6, no. 3, Fall, 1993.

(Russell y Norvig, 1995) Russell, S.J. and Norvig, P. *Artificial Intelligence: A Modern Approach*, Prentice-Hall International Inc., Englewood Cliffs, New Jersey, 1995.

(Sackman, 1974) Sackman, H. *Delphi Assessment : Expert Opinion, Forecasting and Group Process*, Santa Monica, The Rand Corporation, 1974.

(Samuel, 1963) Samuel, A.L. "Some Studies in Machine Learning Using the Game of Checkers." in *Computers and Thought*, E.A. Feigenbaum and J. Feldman (Eds.). McGraw-Hill, New York, pp. 71-105, 1963.

(Scheibler y Schneider, 1985) Scheibler, D. and Schneider, W. "Monte Carlo test of the accuracy of cluster analysis algorithms – a comparison of hierarchical and nonhierarchical methods." *Multiv. Behav. Res.*, vol. 20, pp. 283-304, 1985.

(Scott, 1955) Scott, W.A. "Reliability of Content Analysis: The Case of Nominal Scale Coding." *Public Opinion Quaterly*, XIX, pp. 321-325, 1955.

(Scott et al, 1991) Scott A.C., Clayton, J.E., and Gibson, E.L. *A Practical Guide to Knowledge Acquisition*, Addison-Wesley Publishing Company, Reading, MA, 1991.

(Shafer, 1976) Shafer, G.A. *Mathematical Theory of Evidence*, Princenton University Press, Princenton, New York, 1976.

(Shapiro, 1977) Shapiro, A. "The evaluation of clinical predictions." *New England Journal of Medicine*, vol. 296, pp. 1509-1514, 1977.

(Shaw y Woodward, 1988) Shaw, M. and Woodward, J. "Validation in a knowledge support system. Construing and consistency with multiple experts." *Int. J. Man Machine Studies*, vol. 29, no. 3, pp. 329-350, September 1988.

(Shepard, 1962) Shepard, R. N. "The analysis of proximities: Multidimensional scaling with an unknown distances function (I and II)." *Psychometrika*, vol. 27, pp. 125-139 and 219-246, 1962.

(Shiu et al., 1997) Shiu, S.C., Liu, J.N., and Yeung, D.S. "Formal Description and Verification of Hybrid Rule/Frame-Based Expert Systems." *Expert Systems with Applications*, vol. 13, no. 3, pp. 215-230, October 1997.

(Shooman, 1983) Shoman, M.L. *Software Engineering: Design, Reliability, and Management*. McGraw-Hill, New York, 1983.

(Shortliffe, 1976) Shortliffe, E.H. *Computer-Based Medical Consultation: MYCIN*, American Elsevier, NewYork, 1976.

(Shortliffe et al., 1979) Shortliffe, E.H., Buchanan, B.G., Feigenbaum, E.A. "Knowledge engineering for medical decision making: a review of computer-based clinical decision aids." *Proceedings of the IEEE*, vol. 67, no. 9, pp. 1207-1224, 1979.

(Silva, 85) Silva, M. *Las Redes de Petri en la Automática y la Informática*. Ed. AC, 1985.

(Simon and Herz, 1998) Simon, E. and Herz, M. (17-Jul-98) "Cleanroom Software Architecture." [On-Line]. URL: http://cctr.umkc.edu/user/esimon/457/clean/cleanroom.htm

(Sneath y Sokal, 1973) Sneath, P.H.A., and Sokal, R.R.. *Numerical taxonomy*. W. H. Freeman & Co, San Francisco, 1973.

(Sobey, 1998) Sobey, A. (17-Jul-98) "Software Engineering." [On-Line]. URL: http://louisa.levels.unisa.edu.au/se1/week1/html/index.htm

(Sokal y Sneath, 1963) Sokal, R.R. and Sneath, P.H.A.. *Principles of Numerical Taxonomy*, W. H. Freeman & Co, San Francisco, 1963.

(Spearman, 1904) Spearman, C. "The proof and measurement of association between two things." *American Journal of Psychology*, vol. 15, pp. 71-101, 1904.

(Spearman, 1906) Spearman, C. "A footrule for measuring association." *British Journal of Psychology*, vol. 2, pp. 89-108, 1906.

(Stachowitz y Combs, 1987)* Stachowitz R.A. and Combs J.B. "Validation of Expert Systems." *The Proceedings of the 20th Hawaii International Conference on System Sciences (HICSS)*, vol. 1, pp. 689-695, 1987.

(Stork, 1997a) Stork, D.G. *HAL's Legacy: 2001's computer as dream and reality*, edited by David G. Stork, Foreword by Arthur C. Clarke, MIT Press, 1997

(Stork, 1997b) Stork, D.G. (19-Dec-97) "The end of a era, the beginning of another? HAL, Deep Blue and Kasparov", [On-Line]. URL: http://www.chess.ibm.com/learn/html/e.8.1.html

(Suwa et al., 1982)* Suwa, M., Scott, A.C., Shortliffe, E.H. "An Approach to Verifying Completeness and Consistency in a Rule-Based Expert System." *AI Magazine*, vol. 3, no. 2, pp. 16-21, Fall 1982.

(Taylor et al., 1989) Taylor, R., Porter, D., Hickman, F., Streng, K.H., Tansley, S. and Dorbes, G. *System evolution-principles and methods (the life cycle model).* ESPRIT Project P1098, Deliverable Task G9, Touche Ross.

(Theil, 1970) Theil, H. *Amer. J. Sociol.* vol. 76, pp. 103-154, 1970.

(Torgerson, 1958) Torgerson, W.S. *Theory and Methods of Scaling*, John Wiley and Sons, New York, 1958.

(Tryon, 1939) Tryon, R.C. *Cluster Analysis*. Edwards Brothers, Ann Arbor, Mich. 1939.

(Tucker y Messick, 1963) Tucker, L.R. and Messick, S. "An individual difference model for multidimensional scaling." *Psychometrika*, vol. 28, pp. 333-367, 1963.

(Turing, 1950) Turing, A.M. "Computing Machinery and Intelligence." *Mind*, vol. 59, pp. 433-460.

(Verdaguer et al., 1992) Verdaguer, A., Patak, A., Sancho, J.J., Sierra, C. and Sanz, F. "Validation of the Medical Expert System PNEUMON-IA." *Computers and Biomedical Research*, vol. 25, pp. 511-526, 1992.

(Waterman, 1986) Waterman, D.A. *A Guide to Expert Systems*, Addison-Wesley Publishing Company, 1986.

(Wielinga et al., 1992) Wielinga, B., Schreiber A. Th., and Breuker, J. A. "KADS: A modeling approach to knowledge engineering." *Knowledge Adquisition*, vol. 4, no. 1, pp. 5-53, 1992.

(Wilkins y Buchanan, 1986)* Wilkins, D.C. and Buchanan, B.G. "On Debugging Rule Sets When Reasoning Under Uncertainty." *AAAI-86 Proceedings*, vol. 1, pp. 448-454, 1986.

(Williams, 1976) Williams, G.W. "Comparing the joint agreement of several raters with another rater." *Biometrics*, vol. 32, pp. 619-627, September, 1976.

(Winston, 1984) Winston, P.H. *Artificial Intelligence*, 2nd Ed. Addison-Wesley Publishing Co., 1984.

(Wu y Lee, 1997) Wu, C.H. and Lee, S.J. "Knowledge Verification with an Enhanced High-Level Petri-Net Model." *IEEE Expert*, vol. 12, no. 5, pp. 73-80, Sep/Oct 1997.

(Xu et al., 1992) Xu, L., Krzyzak, A. and Suen, C.Y. "Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition." *IEEE Transactions on Systems, Man and Cybernetics*, vol. 22, no. 3, May/June, 1992.

(Yu et al., 1979) Yu, V.L., Buchanan, B.G., Shortliffe, E.H., Wraith, S.M., Davis, R., Scott, A.C., and Cohen, N.S. "Evaluating the performance of a computer-based consultant." *Computer Programs in Biomedicine*, vol. 9, pp. 95-102, 1979.

(Yule, 1912) Yule, G.U. *J. R. Statist. Soc.*, vol. 75, pp. 579-642, 1912.

(Zelkowitz, 1978) Zelkowitz, M.V. "Perspectives on Software Engineering." *ACM Computing Surveys*, vol. 10, no. 2, pp. 197-216, 1978.

(Zubin, 1938) Zubin, J. *Psychiatry*, vol. 1, pp. 237-247.

Los artículos marcados con un * pueden encontrarse también en:

Gupta, U.G. (Ed.) *Validating and Verifying Knowledge-Based Systems*. IEEE Computer Society Press, Los Alamitos, California, 1991

Los artículos marcados como [On-Line] han sido obtenidos del World Wide Web en la fecha indicada, su disponibilidad en otras fechas no está asegurada.

APÉNDICE A: REGLAS DEL SISTEMA EXPERTO DE PLANIFICACIÓN

```
RULE :  Rule ANA_2 (#1)
If
        Prognosis is TRUE
Then Analysis
        is confirmed.

RULE :  Rule ANA_1 (#2)
If
        Diagnosis is TRUE
Then Analysis
        is confirmed.

RULE :  Rule CHK_ANA_SYN_2 (#3)
If
        Synthesis is TRUE
Then Check_Ana_Syn
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al ser un
problema de sintesis, lo normal es que no sea posible aplicarlo sobre casos historicos
ya que requiere una actuacion sobre el entorno,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo problema Sintesis:
SI => No aplicable a casos historicos,@ADD";)

RULE :  Rule CHK_ANA_SYN_1 (#4)
If
        Analysis is TRUE
Then Check_Ana_Syn
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al tratarse de
un problema de analisis lo podemos aplicar tanto a casos actuales como a casos
historicos,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo problema Analisis:
SI => casos historicos y actuales,@ADD";)

RULE :  Rule CHK_ADEV_3 (#5)
If
        Dev_Phase is "Final"
Then Check_AuxDev
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* En las fases
finales priman mas los aspectos de la validacion orientada al uso. Es necesario tambien
preparar el terreno para una validacion en el entorno de trabajo por parte de los
usuarios"
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Fase de desarrollo:
FINAL => Validaciones orientadas al uso y en el entorno real de trabajo,@ADD";)

RULE :  Rule CHK_ADEV_2 (#6)
If
        Dev_Phase is "Medium"
Then Check_AuxDev
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* En las fases
intermedias el numero de casos de validacion debe ser mayor que en las fases anteriores
y tener una mayor cobertura. En estas fases tambien pueden colaborar expertos ajenos al
de"
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Fase de desarrollo;
MEDIA => Mayor numero de casos y cobertura. Pueden aparecer expertos ajenos al
desarrollo,@ADD";)

RULE :  Rule CHK_ADEV_1 (#7)
If
        Dev_Phase is "Initial"
Then Check_AuxDev
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* En las fases
iniciales se pueden utilizar casos de prueba ya resueltos que analizan aspectos
concretos, siempre dentro de un ambiente de laboratorio. Actuan exclusivamente el Ing.
del conoc. "
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=Fase de desarrollo:
INICIAL => Casos ya resueltos, aspectos concretos, ambiente de laboratorio, llevado a
cabo por el I.C. y el experto colaborador,@ADD";)
```

```
RULE :  Rule CHK_CRI_5 (#8)
If
        Criteria_RS is FALSE
        And Criteria_Experts is TRUE
        And Type_Criteria_Experts is "Consensus"
Then Check_Criteria
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Solo podemos
realizar una validacion contra el experto. La presencia de un consenso entre expertos
facilita la validacion de los resultados del sistema,@ADD";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Podemos
utilizar medidas de pares y ratios de acuerdo,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Validacion contra el
problema: NO,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=Validacion contra el
experto: SI (consenso),@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=Medidas: pares y
ratios,@ADD";)

RULE :  Rule CHK_CRI_4 (#9)
If
        Criteria_RS is FALSE
        And Criteria_Experts is TRUE
        And Type_Criteria_Experts is "Several"
Then Check_Criteria
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Solo podemos
realizar una validacion contra el experto. El estudio con varios expertos es amplio pero
pueden surgir problemas si los expertos difieren mucho entre si. Puede intentarse
constru"
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Podemos
utilizar medidas de pares y medidas de grupo,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Evaluacion contra el
problema: NO,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Evaluacion contra el
experto: SI (grupo de expertos) => mayor objetividad pero problemas si sus diferencias
son elevadas. Tratar de desarrollar un consenso,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Medidas: pares y
grupos,@ADD";)

RULE :  Rule CHK_CRI_3 (#10)
If
        Criteria_RS is FALSE
        And Criteria_Experts is TRUE
        And Type_Criteria_Experts is "One"
Then Check_Criteria
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Solo podemos
realizar una validacion contra el experto. Al contar solo con un experto para la
validacion la objetividad de nuestro estudio puede ponerse en entredicho,@ADD";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Podemos
utilizar medidas de pares y ratios (si consideramos al experto como un estandar),@ADD";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=planred.txt,@TEXT=* Validacion
contra el problema: NO,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Validacion contra el
experto: SI (experto aislado) => puede afectar a la objetividad del estudio,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Medidas: pares y ratios
(si consideramos al experto aislado como un estandar),@ADD";)

RULE :  Rule CHK_CRI_2 (#11)
If
        Criteria_Experts is FALSE
        And Criteria_RS is TRUE
Then Check_Criteria
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* La validacion
solo puede realizarse contra el problema. Al no tener los resultados de expertos humanos
es importante no caer en el error de pretender que el sistema obtenga unos resultados
qu"
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Podemos
utilizar medidas de pares y ratios de acuerdo,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Validacion contra el
problema: SI,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Validacion contra el
experto: NO,@ADD";)
```

```
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Medidas: pares y ratios
de acuerdo,@ADD";)

RULE :  Rule CHK_CRI_1 (#12)
If
        Criteria_Experts is TRUE
        And Criteria_RS is TRUE
Then Check_Criteria
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* La validación
puede realizarse contra el experto o contra el problema. Podemos validar los resultados
de los expertos y compararlos con los del SSEE,@ADD";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Podemos
utilizar medidas de pares, medidas de grupo (con los expertos y la salida real) y ratios
de acuerdo (con la salida real).,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=plared.txt,@TEXT=* Evaluacion contra el
problema: SI,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Evaluacion contra el
experto: SI,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Medidas: pares, grupos
(experto y salida real) y ratios (salida real),@ADD";)

RULE :  Rule CHK_CRI_DOM_2 (#13)
If
        Critical_Domain is FALSE
Then Check_Critical
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@NEW";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al ser un
dominio no critico, los errores no tienen la misma gravedad que en un dominio
critico,@ADD";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=planred.txt,@NEW";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo: No critico =>
Errores de validacion menos peligrosos,@ADD";)

RULE :  Rule CHK_CRI_DOM_1 (#14)
If
        Critical_Domain is TRUE
Then Check_Critical
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@NEW";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* El dominio
critico obliga a una validacion rigurosa ya que un error del sistema puede tener un
coste elevado,@ADD";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=planred.txt,@NEW";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=planred.txt,@TEXT=* Tipo:
Critico => Validacion rigurosa. Los errores en la validacion son peligrosos,@ADD";)

RULE :  Rule CHK_DEV (#15)
If
        Check_AuxDev is TRUE
Then Check_Development
        is confirmed.

RULE :  Rule CHK_DIA_2 (#16)
If
        Diagnosis is FALSE
Then Check_Diagnosis
        is confirmed.

RULE :  Rule CHK_DIA_1 (#17)
If
        Diagnosis is TRUE
Then Check_Diagnosis
        is confirmed.

RULE :  Rule CHK_DOM (#18)
If
        Check_Critical is TRUE
        And Check_Criteria is TRUE
        And Check_Profile is TRUE
Then Check_Domain
        is confirmed.

RULE :  Rule CHK_DPT_1 (#19)
If
        Check_Diagnosis is TRUE
```

```
        And Check_Prognosis is TRUE
        And Check_Therapy is TRUE
Then Check_DPT
        is confirmed.

RULE :  Rule CHK_ENV_4 (#20)
If
        Embedded_System is TRUE
        And Dev_Phase is "Final"
Then Check_Environment
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al estar el
sistema embebido en un entorno mayor es necesario contemplar la validacion de los
interfaces, sobre todo en las etapas mas avanzadas del desarrollo,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Sistema embebido: SI =>
validar los interfaces sobre todo en las etapas finales del desarrollo,@ADD";)

RULE :  Rule CHK_ENV_3 (#21)
If
        Embedded_System is TRUE
        And Dev_Phase is "Medium"
Then Check_Environment
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al estar el
sistema embebido en un entorno mayor es necesario contemplar la validacion de los
interfaces, aunque es mas recomendable hacerlo en etapas mas avanzadas del
desarrollo,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Sistema embebido: SI =>
validar los interfaces pero no en las primeras etapas del desarrollo,@ADD";)

RULE :  Rule CHK_ENV_2 (#22)
If
        Embedded_System is TRUE
        And Dev_Phase is "Initial"
Then Check_Environment
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al estar el
sistema embebido en un entorno mayor es necesario contemplar la validacion de los
interfaces, aunque es mas recomendable hacerlo en etapas mas avanzadas del
desarrollo,@ADD";)
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=planred.txt,@TEXT=* Sistema
embebido: SI => validar los interfaces pero no en las primeras etapas del
desarrollo,@ADD";)

RULE :  Rule CHK_ENV_1 (#23)
If
        Embedded_System is FALSE
Then Check_Environment
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al no estar el
sistema embebido en un entorno mayor la validacion puede centrarse unicamente en el
sistema a desarrollar,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=Sistema embebido: NO =>
validacion independiente del sistema,@ADD";)

RULE :  Rule CHK_NOM_2 (#24)
If
        Results_Nominal is FALSE
Then Check_Nominal
        is confirmed.

RULE :  Rule CHK_NOM_1 (#25)
If
        Results_Nominal is TRUE
Then Check_Nominal
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Para los
resultados nominales no es adecuado el empleo de medidas de asociacion ni del porcentaje
de acuerdo dentro de uno, ya que no preveen ninguna ordenacion en sus categorias,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo resultados: NOMINAL
=> no medidas de asociacion ni porcentajes dentro de uno,@ADD";)

RULE :  Rule CHK_NUM_2 (#26)
If
        Results_Numeric is FALSE
Then Check_Numeric
        is confirmed.
```

```
RULE :  Rule CHK_NUM_1 (#27)
If
        Results_Numeric is TRUE
Then Check_Numeric
        is confirmed.

RULE :  Rule CHK_ORD_2 (#28)
If
        Results_Ordinal is FALSE
Then Check_Ordinal
        is confirmed.

RULE :  Rule CHK_ORD_1 (#29)
If
        Results_Ordinal is TRUE
Then Check_Ordinal
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Para los
resultados ordinales es necesario tener en cuenta la distancia existente entre sus
categorias con medidas como kappa ponderada o el porcen. de acuerdo dentro de uno.
Tambien son apli"
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo resultados: ORDINAL
=> hay que tener en cuenta el orden de las categorias. Se recomienda utilizar kappa
ponderada, porcentajes dentro de uno y tambien medidas de asociacion,@ADD";)

RULE :  Rule CHK_PLAN (#30)
If
        Check_Domain is TRUE
        And Check_System is TRUE
        And Check_Development is TRUE
Then Check_Plan
        is confirmed.

RULE :  Rule CHK_PRO_2 (#31)
If
        Results_Probabilistic is FALSE
Then Check_Probabilistic
        is confirmed.

RULE :  Rule CHK_PRO_1 (#32)
If
        Results_Probabilistic is TRUE
Then Check_Probabilistic
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Los resultados
que forman un vector probabilistico puede compararse a traves de distancias aritmeticas
(no incluidas en SHIVA),@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo resultados: VECTOR
PROBABILISTICO => utilizar distancias aritmeticas (no en SHIVA),@ADD";)

RULE :  Rule CHK_PRO_2 (#33)
If
        User_Profile is "Non Expert"
Then Check_Profile
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Los usuarios no
expertos son adecuados solo para una validacion orientada al uso,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo usuarios: NO
EXPERTOS => Colaboran solo en la validacion orientada al uso,@ADD";)

RULE :  Rule CHK_PRO_1 (#34)
If
        User_Profile is "Expert"
Then Check_Profile
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Los usuarios
expertos pueden colaborar activamente en la validacion de resultados a traves de tests
de campo,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo usuarios: EXPERTOS
=> Validacion orientada a los resultados con tests de campo,@ADD";)

RULE :  Rule CHK_PRO_3 (#35)
If
        Prognosis is FALSE
Then Check_Prognosis
        is confirmed.
```

```
RULE :  Rule CHK_PRO_2 (#36)
If
        Prognosis is TRUE
        And Criteria_RS is TRUE
Then Check_Prognosis
        is confirmed.

RULE :  Rule CHK_PRO_1 (#37)
If
        Prognosis is TRUE
        And Criteria_RS is FALSE
Then Check_Prognosis
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* En los
problemas de pronostico generalmente se suele contar con una solucion real (el resultado
final del pronostico), por lo que seria interesante tratar de hallarla para mejorar
nuestro pro"
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo Analisis:
Pronostico => Para mejorar el proceso de validacion puede tratar de hallar la solucion
real del pronostico,@ADD";)

RULE :  Rule CHK_SUB_2 (#38)
If
        Ind_Modules is FALSE
Then Check_Subsystems
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al no poder
subdividir el sistema en modulos independientes debemos validarlo como un todo, lo que
puede dificultar la deteccion de errores,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Modulos independientes:
NO => Validar al sistema como un todo,@ADD";)

RULE :  Rule CHK_SUB_1 (#39)
If
        Ind_Modules is TRUE
Then Check_Subsystems
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al estar el
sistema dividido en modulos mas o menos independientes, podemos realizar la validacion
de cada uno de ellos de forma independiente,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Modulos independientes:
SI => Podemos realizar validaciones independientes,@ADD";)

RULE :  Rule CHK_SYS_1 (#40)
If
        Check_Subsystems is TRUE
        And Check_Uncertainty is TRUE
        And Check_Type_Problem is TRUE
        And Check_Type_Output is TRUE
        And Check_Environment is TRUE
Then Check_System
        is confirmed.

RULE :  Rule CHK_THE_3 (#41)
If
        Therapy is FALSE
Then Check_Therapy
        is confirmed.

RULE :  Rule CHK_THE_2 (#42)
If
        Therapy is TRUE
        And Critical_Domain is FALSE
Then Check_Therapy
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al no ser un
dominio critico se puede aplicar la terapia y comprobar si los resultados obtenidos son
correctos,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo Sintesis: Terapia
=> La terapia puede aplicarse al no ser un dominio critico,@ADD";)

RULE :  Rule CHK_THE_1 (#43)
If
        Therapy is TRUE
        And Critical_Domain is TRUE
Then Check_Therapy
```

```
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* Al ser un
dominio critico la terapia recomendada por el sistema solo se podra aplicar en aquellos
casos en los que coincida con expertos humanos,@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Tipo Sintesis: Terapia
=> La terapia en un dominio critico solo puede aplicarse si coincide con la opinion de
un experto,@ADD";)

RULE :  Rule CHK_TYP_OUT (#44)
If
        Check_Nominal is TRUE
        And Check_Ordinal is TRUE
        And Check_Numeric is TRUE
        And Check_Probabilistic is TRUE
Then Check_Type_Output
        is confirmed.

RULE :  Rule CHK_TYP_PRO_1 (#45)
If
        Check_Ana_Syn is TRUE
        And Check_DPT is TRUE
Then Check_Type_Problem
        is confirmed.

RULE :  Rule CHK_UNC_2 (#46)
If
        Uncertainty is FALSE
Then Check_Uncertainty
        is confirmed.
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Incertidumbre:
NO,@ADD";)

RULE :  Rule CHK_UNC_1 (#47)
If
        Uncertainty is TRUE
Then Check_Uncertainty
        is confirmed.
        And Execute "WriteTo"(@STRING="@TRANSCRIPT,@FILE=plan.txt,@TEXT=* SHIVA supone
resultados categoricos que no se ven afectados por la incertidumbre. Esta puede ser
analizada con medidas de exactitud o analisis de sensibilidad (no incluidos en
SHIVA),@ADD";)
        And Execute "WriteTo"(@STRING="@FILE=planred.txt,@TEXT=* Incertidumbre: SI =>
Analizarla con medidas de exactitud o analisis de sensibilidad (no en SHIVA),@ADD";)

RULE :  Rule SYN_1 (#48)
If
        Therapy is TRUE
Then Synthesis
        is confirmed.
```

APÉNDICE B: ARTÍCULOS PUBLICADOS

IEEE Trans. on Information Technology in Biomedicine

Moret-Bonillo, V., Mosqueira-Rey, E. and Alonso-Betanzos, A. "Information Analysis and Validation of Intelligent Monitoring Systems in Intensive Care Units." *IEEE Transactions on Information Technology in Biomedicine*, vol. 1, no. 2, pp. 87-99, June 1997.

IEEE TRANSACTIONS ON

A PUBLICATION OF THE IEEE ENGINEERING IN MEDICINE AND BIOLOGY SOCIETY

JUNE 1997 VOLUME 1 NUMBER 2 ITIBFX (ISSN 1089–7771)

Information Analysis and Validation of Intelligent Monitoring Systems in Intensive Care Units

Vicente Moret-Bonillo, *Member, IEEE*, Eduardo Mosqueira-Rey, and Amparo Alonso-Betanzos, *Member, IEEE*

Abstract—**Validation of intelligent systems is an important task to perform. Typically the results of the validation analysis are used to verify whether or not the system satisfies the initial design requirements, and to acquire new knowledge and/or refine the knowledge already acquired. In practice, the validation of intelligent systems usually requires the application of several different techniques (e.g., retrospective, prospective, quantitative). In this work the authors present the methodology devised to validate PATRICIA: an intelligent monitoring system designed to advise clinicians on the management of patients dependent on mechanical ventilation. The application of this methodology requires that appropriate validation paradigms are selected, depending on both the application domain and the characteristics of the intelligent system. The article also presents and discusses validation results.**

Index Terms— **Evaluation of intelligent systems, intelligent monitoring, performance analysis, validation.**

I. Introduction

THE construction of any software system is a cyclical process that involves specification of requirements, design and construction (initial prototyping, refined prototype, and final product), performance analysis, and feedback of results for further refinements. In addition, knowledge engineering techniques requires procedures for knowledge elicitation, concepts formalization and knowledge operationalization [1]. The complete intelligent system should be structured within a well-defined and efficient architecture (i.e., with multiple knowledge bases, modular design, well-established distinction between declarative and procedural knowledge, etc.), and finally, the performance of the whole product should be tested.

Analysing the performance of intelligent systems is a process that involves multiple tasks [2], and the techniques usually employed to carry out this analysis should be based on methodological approaches [3]. For reasons which will become evident, these approaches should be based on classical software engineering procedures, which are adapted to the differential characteristics of intelligent systems [4].

A classical approach to the analysis of performance of intelligent systems involves the following phases [2]: *software verification*, that is, testing the consistency, completeness, and appropriateness of the software [5]; *system validation*, that is, determining the suitability of the final program with respect to both user needs and initial requirements and specifications [5]; *credibility analysis*, that is, the extent both to which a system is acceptable to users and to which users can have confidence in it; *assessment*, the objective of which is to evaluate to what extent system and user can cooperate beyond the correctness of the decisions that the system makes; and finally, *evaluation*, which is generally concerned with cost-benefit analysis. All of the above-mentioned phases are hierarchically ordered in such a way that any inconsistency at a lower level leads to incoherencies at an upper level.

The two first phases in the programme mentioned above (i.e., verification and validation) are also known as "the V&V phase" which represents the most crucial stage in ensuring quality in the system. We can hardly implement the credibility, assessment and evaluation phases if a given system is not firstly verified and validated adequately.

A. Influence of the Application Domain in the Validation Strategy

Not all domains require the same treatment in the validation process since there are some domains for which it is always possible to reconsider decisions and conversely, there are other domains where once a given decision has been undertaken, it is not possible to reconsider it due to its consequences. We will call these latter domains "*critical domains*," where the reasoning processes and results are both highly dependent on the characteristics of the environment as well as on the complexity of the overall decision process.

In medicine a good example of a critical domain is the intensive care unit (ICU), where the working environment is defined by the following four different generic elements: patient, clinical personnel, therapeutic equipment, and monitoring equipment.

In the ICU all relevant information on the patient can be obtained from the patient's clinical history, patient symptoms, clinical observations, bedside monitors, and laboratory results. The information from these sources is brought together by clinical personnel in order to make inferences about the patient's pathophysiological condition and evolution, and to prescribe appropriate therapies. In this context, for example, appropriate therapies may include drug administration, adjustments to electrical/electronic and/or mechanical devices, and manual intervention. The final result is a series of actions that tend to control ventilation and oxygenation of the patient, nutritional, electrolytic and acid-base balances, neurophysiological activity, and patient hemodynamics [6]. As part of

Manuscript received January 3, 1997; revised March 25, 1997. This work was supported in part by the Spanish CICyT under Projects TIC 91-0789 and TIC 96-0590, and by the Xunta de Galicia, "Programa de Becas del Tercer Ciclo", DOG-245.

The authors are with the Laboratory for R&D in Artificial Intelligence (LIDIA), Department of Computer Science, University of La Coruña, 15071 La Coruña, Spain (e-mail: civmoret@udc.es; eduardo@udc.es; ciamparo@udc.es).

Publisher Item Identifier S 1089-7771(97)05616-1.

the overall monitoring process, time constraints are crucial, due to—at least in part—the different persistence or latency of the variables, and the different monitoring and action priorities to be taken into account, depending on the patient's pathophysiological condition.

Given the above-mentioned factors, it can be said that the optimal performance of an intelligent system designed to help clinicians in the ICU relies greatly on the validation process. The system should be carefully tested, taking into account initial design specifications, in order to guarantee appropriate interpretations and therapies. In addition, the system should become part of the working environment without substantially altering clinical routine.

II. An Overview of Some Existing Validation Paradigms

As previously mentioned, the term "validation" refers to the process that permits the establishment of whether or not a given intelligent system satisfies initial design requirements while, simultaneously taking into account the restrictions of the working environment. In the literature several different methods for performing this task are described. Each method can be evaluated according to the following criteria: the standards used in the validation process, the qualitative or quantitative nature of the method, and the retrospective or prospective nature of the method. These three criteria define our validation paradigms.

A. Gold-Standard versus Non-Gold-Standard Validation Methods

Traditional approaches to the validation of intelligent systems usually endeavour to compare the results of the system with a "gold-standard" [7]. However, in critical domains, this gold-standard is not always easy to establish. For example, in the ICU, the opinions of experts, even when confronted with the same clinical problem, can vary. This variation in opinion can be accounted for in the following ways: 1) medicine is basically a nondeterministic science, 2) stress can affect the clinical interpretation process, 3) equivalent but not identical actions can be undertaken in trying to resolve the same problem, and 4) priorities are imposed in the ICU by time constraints and by the relative urgency of a given crisis. Thus, even if there are one or more "gold-experts" we cannot be sure that interpretations and decisions follow "gold-standard principles."

In situations where a gold standard cannot be established, methods such as *Williams measurements* [8] and *Cluster analysis* [9], [10], facilitate the validation process. These methods are based on comparison of results supplied by the system with those supplied by a group of experts, when each is independently faced with the same set of problem data. These methods have as their objective an analysis of the extent of agreement between experts and the system and so to locate the system within the group of experts.

Williams measurements are useful to investigate how much a single isolated expert agrees with a group of experts, and to what extent this agreement is similar to agreement among members of the group. The Williams method defines an index of comparison (I_n) which is a ratio between on the one hand, the agreement between the isolated expert and the reference group of experts (P_0) and on the other hand, the observed overall agreement within the group (P_n). The equations defining these concepts are the following:

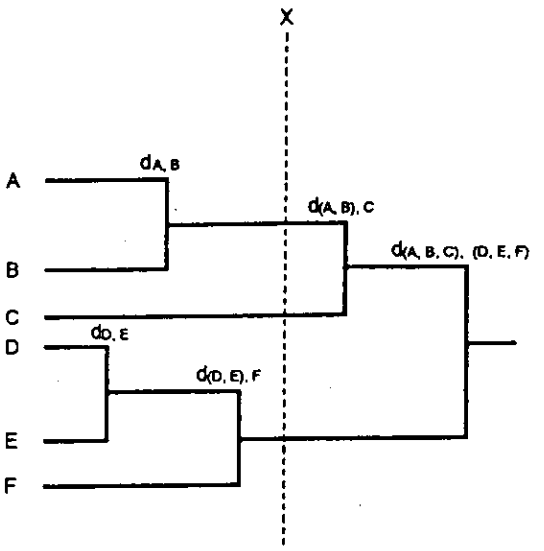


Fig. 1. Cluster analysis tree or "dendrogram" with di, j representing the distance between expert i and expert j. Splitting the dendrogram at line X we obtain three clusters (A, B), (C), and (D, E, F).

$$I_n = \frac{P_0}{P_n} \tag{1}$$

$$P_0 = \frac{\sum_{a=1}^{n} P_{(0,a)}}{n} \tag{2}$$

$$P_n = \frac{2\sum_{a=1}^{n-1}\sum_{b=a+1}^{n} P_{(a,b)}}{[n(n-1)]} \tag{3}$$

where $P_{(a,b)}$ represents percentage agreement between expert a and expert b, n the number of experts (not including the isolated expert), and 0 the isolated expert.

The range both of P_0 and P_n is $[0, 1]$ while the values of I_n can be interpreted as follows.

- If $0 \leq I_n < 1$, the agreement between the isolated expert and the group of experts is lower than the agreement among members of the group.
- If $I_n = 1$, the agreement between the isolated expert and the group of experts is equal to the agreement among members of the group.
- If $I_n > 1$, the agreement between the isolated expert and the group of experts is greater than the agreement among members of the group.

Another possible method to use when a gold standard cannot be established is cluster analysis. This method endeavours to establish "categories" of experts and to identify the level of "expertise" to which our system belongs. Cluster methods are classified into two main types: hierarchical and nonhierarchical [11].

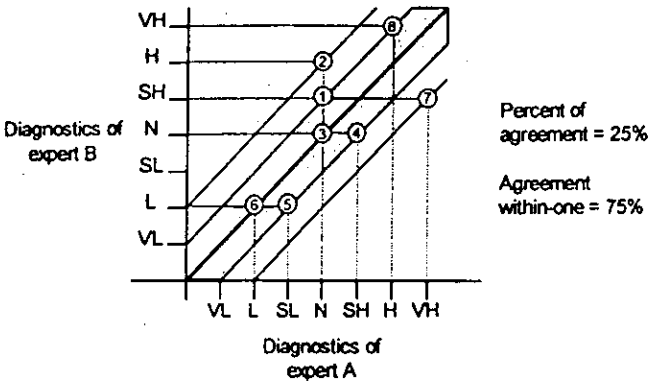
Hierarchical methods are based on the construction of an agreement matrix that describes distances between all

Semantic scale for the symbolic classification of a given variable.	Very Low (VL)	Low (L)	Slightly Low (SL)	Normal (N)	Slightly High (SH)	High (H)	Very High (VH)

(a)

Cases	Expert A	Expert B
1	Normal	Slightly High
2	Normal	High
3	Normal	Normal
4	Slightly High	Normal
5	Slightly Low	Low
6	Low	Low
7	Very High	Slightly High
8	High	Very High

(b)



(c)

Fig. 2. (a) Semantic scale for the symbolic classification of a given diagnosis. (b) Two experts interpretations for a given diagnosis. (c) Representation of deviations from those interpretations. The shadowed area represents within-one agreement, and the numbers in the figure represent the cases referred to in (b).

A\B	Very Low	Low	Slightly Low	Normal	Slightly High	High	Very High	Σ→
Very Low	0 (0)	0	0	0	0	0	0	0
Low	0	.125 (.031)	0	0	0	0	0	.125
Slightly Low	0	.125	0 (0)	0	0	0	0	.125
Normal	0	0	0	.125 (.094)	.125	.125	0	.375
Slightly High	0	0	0	.125	0 (.031)	0	0	.125
High	0	0	0	0	0	0 (.016)	.125	.125
Very High	0	0	0	0	.125	0	0 (.016)	.125
Σ↓	0	.25	0	.25	.25	.125	.125	1

$p_c = .188$
$p_o = .25$
$\kappa = .077$

Fig. 3. Calculation of p_c: (a) Calculation of marginal proportions summing the observed proportions for the row and column of each cell, (b) the multiplication of the marginal proportions, representing the proportion of agreement expected by chance for each cell (between brackets), and (c) the summing of all the proportions expected by chance on the agreement diagonal.

members of the study. This matrix can be built, for example, using percentage agreement between experts. Constructing this matrix, we follow a sequence of nested groupings, thus obtaining a tree structure, called a dendrogram, where each level of aggregation represents a partition of the given set of members (Fig. 1).

Non-hierarchical methods on the other hand, create a division that minimizes the sum of squares of the distances between each point and the centroid of its class (several distances can be used: Euclidean, Chebychev, etc). In this case the method consists of performing an initial subdivision into a predefined number of classes that is improved by iteration, shifting the elements of one cluster to another and calculating, after each assignment, the new centroids. Optimal configuration is obtained when in the process of a complete iteration, no element switches from one cluster to another.

B. Qualitative versus Quantitative Validation Methods

Another different aspect of the validation problem concerns the nature both of the method and of the information used (i.e., qualitative or quantitative).

Qualitative approaches are based on the application of subjective techniques of evaluation [12]. Some of the best known qualitative methods are the *Surface Validation Method*, the *Turing Test*, and the *Field Test*.

The "Surface Validation Method" consists of an informal approach whereby knowledge engineers and domain experts

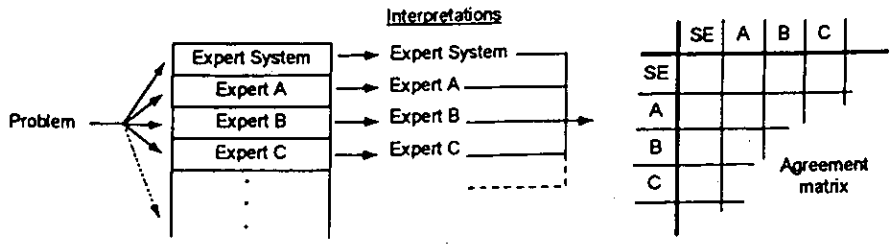


Fig. 4. Retrospective validation scheme.

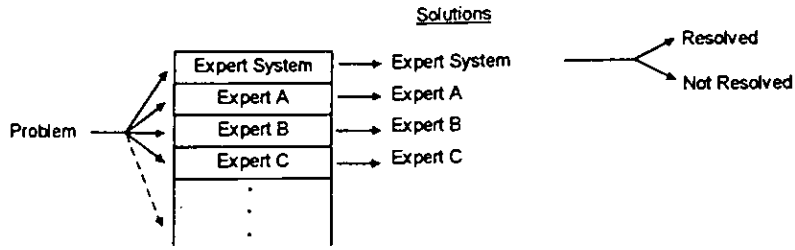


Fig. 5. Prospective validation scheme.

discuss and analyze the validity of conclusions obtained by the system. This technique can be useful in evaluating a given module during the system's development phase, but it cannot be used to investigate the performance of the whole system because it does not follow a formal approach [12].

The "Turing Test" consists of a "multiple hidden evaluation technique." The expert appointed to test the system (i.e., the "evaluator") is presented with a number of sets of data that have been independently interpreted both by the intelligent system and by a number of experts. Data and interpretations are presented to the evaluator using the same format, in such a way that he or she can identify neither the intelligent system nor any of the experts [2]. Because of its particular characteristics some authors recommend this method for validating medical expert systems [13].

The "Field Test" allows qualitative validation of intelligent systems in their working environment. This method transfers the responsibility of the validation process to the end-user who is required to stress the system in order to detect inconsistencies in its reasoning processes and other problems that can only be detected in the operational field [14].

Quantitative validation methods, on the other hand, are based on statistical analysis which compares the conclusions of the intelligent system with those of domain experts. Some of the best known quantitative methods are agreement measurements such as percent agreement and within-one percentage agreement [15], or kappa and weighted kappa measurements [16].

The percent agreement technique is a very simple one where an index of agreement is obtained by dividing the number of cases in which both the intelligent system and the domain expert agree, by the total number of cases. Given a system in which diagnoses are divided into categories denoting a given semantic set, we can define this measurement as

$$\text{Percentage agreement} = \frac{\sum_{i=j}^{k} n_{ij}}{N} \qquad (4)$$

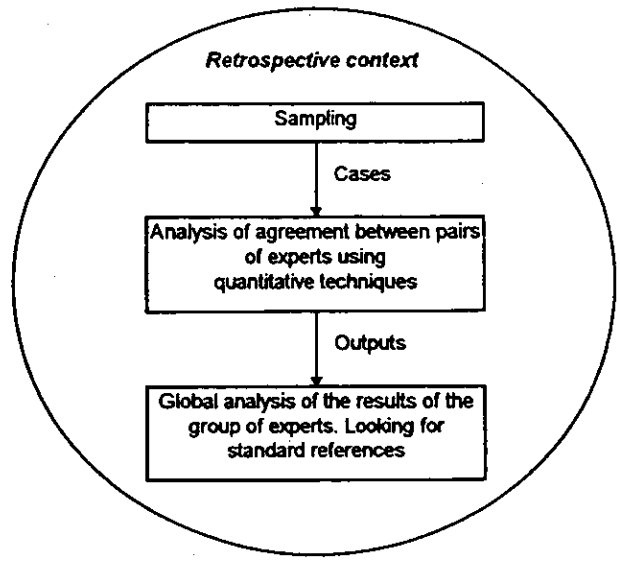


Fig. 6. Representation of the proposed validation method.

where N represents the total number of cases under consideration, k the number of categories into which the diagnosis under consideration is divided, and n_{ij} the number of cases in which the first expert diagnosed category i and the second category j (thus, n_{ij} where $i = j$ represents the number of cases where exact agreement occurs).

Within-one agreement measurements are based on the same principle, but they are specifically indicated in cases where interpretations fit into a previously defined semantic set, and where linguistic nuances are taken into account. The interpretations expert—intelligent system (or any other "pair" under consideration) differing in just one linguistic category are considered in partial agreement. This measurement can be defined as

$$\text{Within one percentage agreement} = \frac{\sum_{i=j\pm1}^{k} n_{ij}}{N} \qquad (5)$$

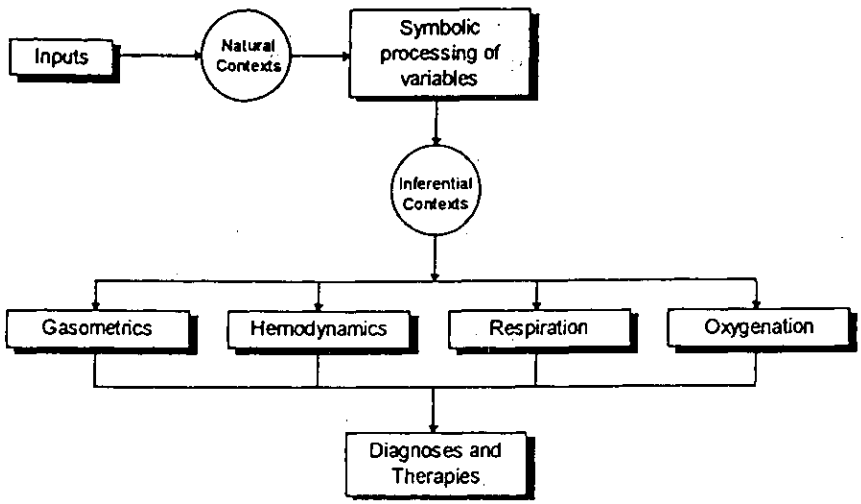


Fig. 7. Distribution of procedural knowledge in PATRICIA.

where N, k, and n_{ij} are the same parameters as in (4). How this measurement of agreement operates in a hypothetical case is shown in Fig. 2 [17].

As previously mentioned, other quantitative validation methods are based on kappa measurements. Kappa measurements were developed by Cohen in an attempt to eliminate those agreements expected by chance [16]. The kappa index is defined as

$$\kappa = \frac{p_o - p_c}{1 - p_c} \tag{6}$$

where p_o is the observed proportion of agreement, and p_c is the proportion of agreement expected by chance. This latter proportion is found by summing along the agreement diagonal the product of the marginal proportions for the row and column of each cell of that diagonal. This calculation is illustrated in Fig. 3, and is based on the same data as in Fig. 2.

Kappa measurements treat all discrepancies in exactly the same way. However, in clinical practice it is not the same, for a patient suffering "severe hypertension," to say that he or she is suffering "moderate hypertension" or "severe hypotension."

In order to avoid this inconsistency in the model, Cohen defined a new index called the "weighted kappa measurement" which weights disagreements (such as described above) according to their relative importance. Weighted kappa is defined as

$$\kappa_w = 1 - \frac{\sum v_{ij} p_{oij}}{\sum v_{ij} p_{cij}} \tag{7}$$

where p_{oij} is the proportion observed in the cell ij, p_{cij} the proportion expected by chance in the cell ij and v_{ij} the disagreement weight of that cell (a weight that represents the relative importance of the disagreement).

C. Retrospective versus Prospective Validation Methods

The aim of retrospective validation is to investigate the extent of similarity between the conclusions of the intelligent system and those of either the domain experts or the gold standard. The method consists of the construction of an agreement matrix and, once this is done, the evaluation of whether or not the intelligent system agrees with the domain experts to the same extent that the experts agree among themselves (see Fig. 4).

Prospective validation on the other hand, is an "against the problem" procedure in that it is necessary to confirm that the decisions made by the intelligent system resolved the problem in question (or if the interpretations are correct) [14]. Fig. 5 illustrates the prospective validation method. In practice, and for obvious reasons, this method can only be applied in those cases where the suggestions of the intelligent system coincide exactly with the gold standard. This fact can have negative effects in relation with, for example, the number of sampling data used in the validation process.

The nonexistence of gold standards in critical domains makes prospective validation methods generally impractical. However, these validation methods do have a use in systems that include prediction facilities and for which a gold standard is available.

III. A Strategy for the Validation of Intelligent Systems in Critical Domains

The choice of which validation technique to be used depends basically on the problem domain and on the characteristics of the intelligent system under consideration. In the first place, the application domain can limit possibilities of election between different validation paradigms in the following ways.

- If the domain is a critical one we can never run the risk of an erroneous decision, and thus, prospective validation methods are not appropriate;
- There are some domains for which it is difficult to establish gold standards. In these cases, our validation approach should be constructed without gold standard considerations;
- If the output of the system is a set of diagnoses which are categorically and linguistically labeled, then we can use quantitative measurements (e.g., percentage agreement, kappa and weighted kappa measurements) since

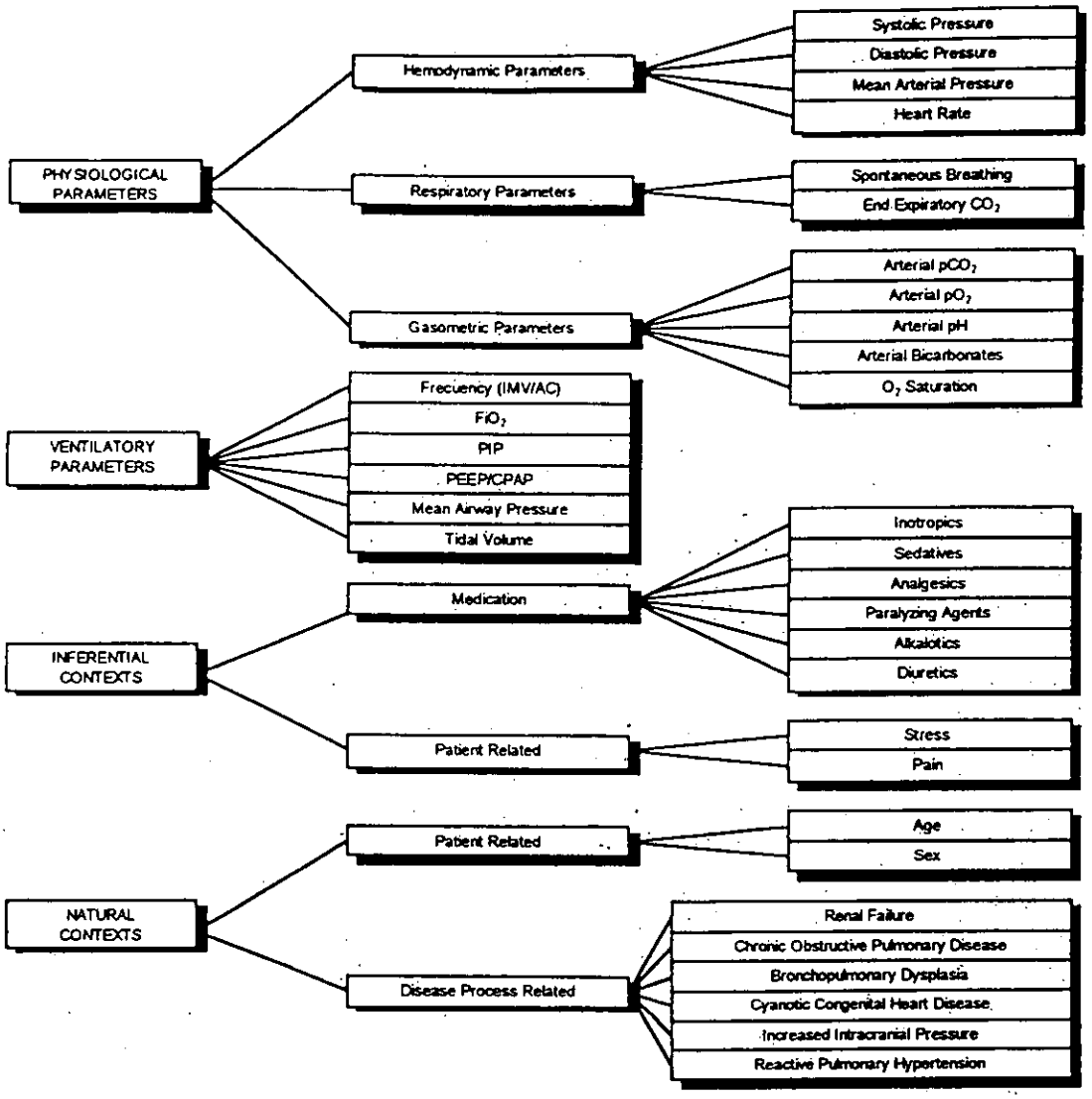


Fig. 8. An example of declarative knowledge in PATRICIA.

the construction of a contingency table to facilitate the application of such methods is comparatively simple.

Secondly, the particular characteristics of an intelligent system also play a part in the selection of validation paradigms. Thus, if the system can be divided into a series of subsystems, then each unit can be analyzed separately, as long as there are no modules that act as input–output for other modules. In such a situation, the validation process is complicated by the fact that the results of one module depend on the results of another and, as a consequence, it may be difficult to identify the source of an error. Moreover, if the input manipulated by the system includes numerically presented uncertain knowledge, then it may be difficult to establish the numeric values representing such an uncertainty. For this reason, it may be necessary to make systematic changes within the system in order to evaluate the effect of these factors on the system's responses.

The methodology described below is particularly suited for critical domains where, although there is not a standard reference, the opinions of several experts are available. These expert opinions should be based on a statistically representative number of cases, for which there are a sufficient quantity of data. In addition, system output should be hierarchically arranged in a semantic set which takes into account linguistic nuances (i.e. very, slightly, etc.) Furthermore, the system's interpretations should not be affected by certainty factors or any other imprecise statistical measurement, since the methodology will ignore them.

The following methodology and corresponding phases can be applied to any system and domain complying with the above criteria.

- **Sampling**: For an accurate analysis of system performance it is essential to have available an adequate quantity of data relating to clinical cases. Also important is the coverage of the data, in that the relative frequency of the problems dealt with by the system is reflected in the cases [13].
- **Analysis of agreement between pairs of experts**: For each module and each diagnostic/therapeutic category, an analysis of agreement between pairs of experts (including the intelligent system) is proposed, using percentage

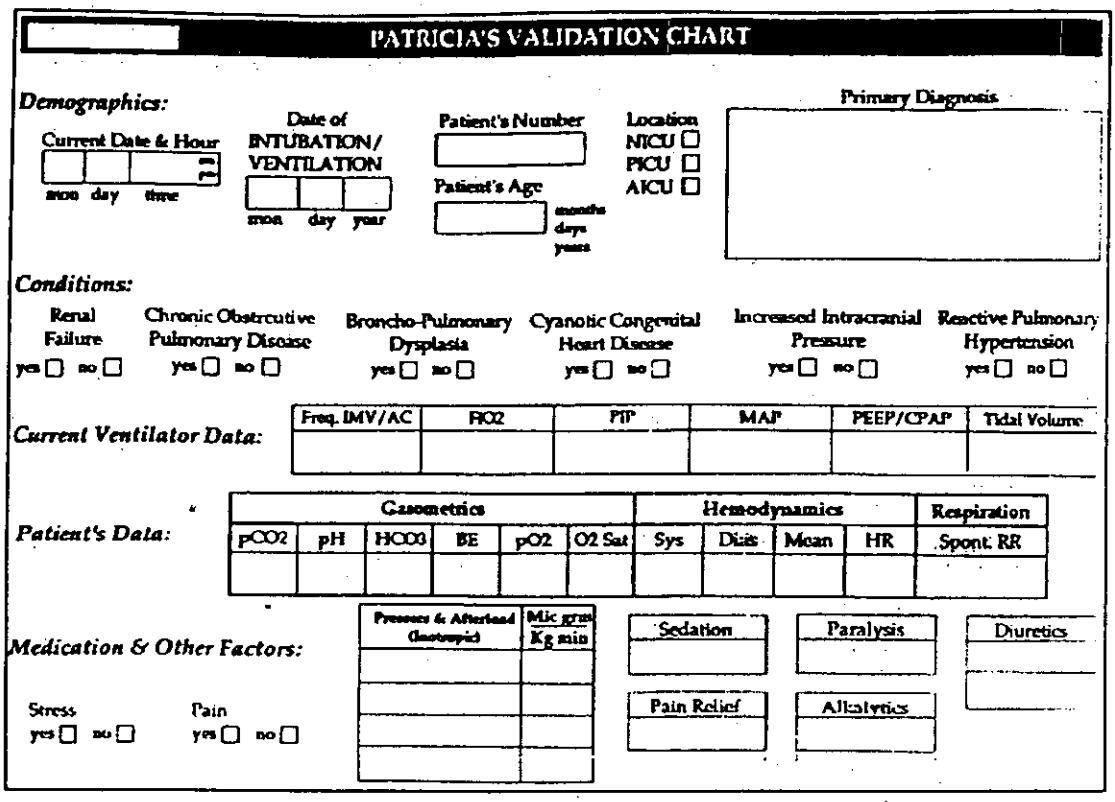


Fig. 9. Patient data in the validation chart.

agreement, within-one percentage agreement, kappa and weighted kappa measurements, with the object of covering different aspects of the global problem.

- **Analysis of agreement between experts**: Using the results obtained for pairs of experts, the use of Williams measurements is recommended in order to determine to what extent agreement between system and group of experts deviates from agreement among experts. Cluster analysis may also be useful in grouping expert and system results in classes.

In this way the methodology can be fitted into a retrospective context, whereby a series of quantitative techniques can be used to measure agreement between experts. The subsequent results are used as input for other quantitative methods (Williams measurements and cluster analysis), which, analysing the results of all experts together, can endeavour to establish a standard reference (Fig. 6).

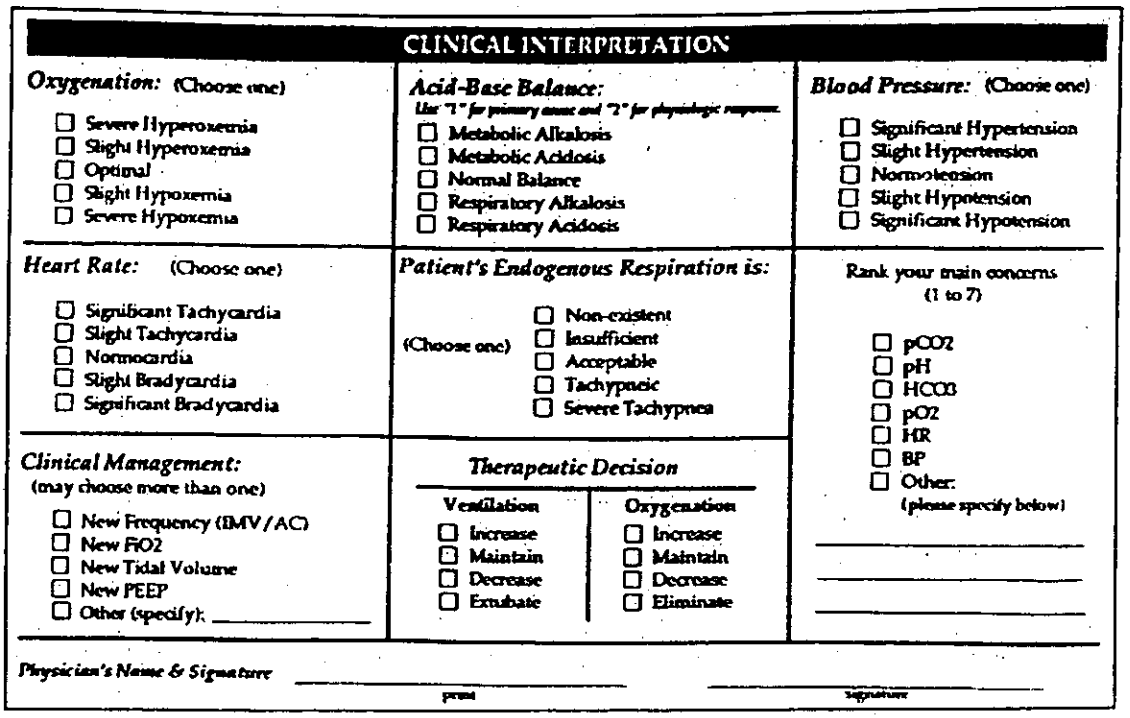
In order to apply this methodology, and given that within-one percentage agreement is used, it is necessary to form a scale of the linguistic labels given to each diagnostic/therapeutic category. This makes it possible to discriminate linguistic nuances and at the same time identify overlapping between categories.

It may happen, on the other hand, that in using kappa measurements, the values obtained are low even though agreement between experts is high [18]. This is a limitation in the technique, caused fundamentally by the characteristics of the sample (i.e., a sample not entirely representative of the widest possible spectrum of cases).

Finally, for the implementation of the cluster analysis technique, the use of hierarchical, rather than nonhierarchical methods is recommended. This is because, firstly results converge more rapidly and secondly, there is no need to specify in advance the number of clusters to be obtained. With nonhierarchical methods on the other hand, in order to be able to use an iterative algorithm, it is necessary to work with the coordinates that characterize the expert (for example, the expert's diagnoses for each one of the categories, then assign a number to each diagnosis and finally, treat the expert as a numerical vector), all this in order to compare distances between those experts. This technique does have some drawbacks such as the loss of the possibility to use kappa measurements or percentage agreements as distances between experts, and/or the fact that it is an unnatural way to work with semantic elements. Since the allocation of an expert to a cluster cannot be changed, the problem with the hierarchical methods is that, the resulting categories may be different to those obtained by using nonhierarchical methods.

IV. Validating the Intelligent Monitoring System PATRICIA

The intelligent monitoring system PATRICIA was designed to assist clinical personnel in the supervision of intensive care patients dependent on mechanical ventilation. Fundamentally, PATRICIA's function is to classify the patient's respiration, oxygenation, acid-base balance and hemodynamics; in order to facilitate appropriate diagnoses and therapies. To do this, contextual information from the patient (natural contexts) and/or



CLINICAL INTERPRETATION

Oxygenation: (Choose one)
- ☐ Severe Hyperoxemia
- ☐ Slight Hyperoxemia
- ☐ Optimal
- ☐ Slight Hypoxemia
- ☐ Severe Hypoxemia

Acid-Base Balance:
Use "1" for primary cause and "2" for physiologic response.
- ☐ Metabolic Alkalosis
- ☐ Metabolic Acidosis
- ☐ Normal Balance
- ☐ Respiratory Alkalosis
- ☐ Respiratory Acidosis

Blood Pressure: (Choose one)
- ☐ Significant Hypertension
- ☐ Slight Hypertension
- ☐ Normotension
- ☐ Slight Hypotension
- ☐ Significant Hypotension

Heart Rate: (Choose one)
- ☐ Significant Tachycardia
- ☐ Slight Tachycardia
- ☐ Normocardia
- ☐ Slight Bradycardia
- ☐ Significant Bradycardia

Patient's Endogenous Respiration is:
(Choose one)
- ☐ Non-existent
- ☐ Insufficient
- ☐ Acceptable
- ☐ Tachypneic
- ☐ Severe Tachypnea

Rank your main concerns (1 to 7)
- ☐ pCO2
- ☐ pH
- ☐ HCO3
- ☐ pO2
- ☐ HR
- ☐ BP
- ☐ Other: (please specify below)

Clinical Management:
(may choose more than one)
- ☐ New Frequency (IMV/AC)
- ☐ New FiO2
- ☐ New Tidal Volume
- ☐ New PEEP
- ☐ Other (specify): _____

Therapeutic Decision

Ventilation	Oxygenation
☐ Increase	☐ Increase
☐ Maintain	☐ Maintain
☐ Decrease	☐ Decrease
☐ Extubate	☐ Eliminate

Physician's Name & Signature _____ (print) _____ (signature)

Fig. 10. Clinical interpretations in the validation chart.

contextual information from the process (inferential contexts) are used [19]. As a knowledge representation scheme, PATRICIA uses a mixed strategy where declarative knowledge is represented by classes that follow an object-oriented approach, and dynamic knowledge is distributed as a series of production rules, organized into five rule bases, each one dedicated to a specific aspect of the global problem in question. Fig. 7 illustrates the modular architecture of procedural knowledge and Fig. 8, an example of declarative knowledge.

The system architecture includes three modules: a *deterministic module* for the symbolic classification of numerical parameters within a semantic space previously defined for each variable, with the classification criteria being defined by the expert and adapted to the case under consideration [20], a *heuristic module*, which makes diagnoses and prescribes therapies, both based on results obtained from the deterministic module, and finally a *control module* which includes a temporal strategy that prioritises tasks and controls the global supervision process. A detailed description of PATRICIA has been published elsewhere [21].

PATRICIA has been implemented using "Nexpert Object" knowledge engineering tools, and runs on UNIX workstations as well as on PC's with "Windows" platforms.

From a validation perspective, PATRICIA is characterized as follows:

- it is designed to assist doctors responsible for patients dependent for survival on mechanical ventilation. Thus, it functions within a critical domain;
- because its architecture is modular, global validation is only possible via the validation of each of its modules separately;
- inputs to the system consist of a series of demographic, ventilatory and physiological parameters, a series of natural contexts which affect the symbolic processing of the parameters, and a series of inferential contexts which influence diagnoses and therapies;
- the set of interpretations produced by PATRICIA is performed by a series of linguistic labels, arranged hierarchically [22];
- not just one therapy is recommended; rather, various possible therapies are suggested, each one different but at the same time, from a physiological point of view, equivalent.

In order to carry out validation of PATRICIA, a sample was obtained of 30 actual clinical cases, representative of the most typical situations in the system's application domain. Each one of this 30 patients was monitored over time, so as to obtain for each case, different sets of data corresponding to different moments in time and different medical conditions. All this information was assembled and recorded in charts such as that reproduced in Fig. 9. The minimum number of sets of data for each patient was two and the maximum was eight.

For each set of data (i.e. for each chart), seven diagnostic/therapeutic categories (oxygenation, acid-base balance, arterial pressure, heart rate, endogenous respiration, oxygenatory therapy and ventilatory therapy) were analyzed. The global sample consisted in 1470 numerical data. These data were analyzed "blind" and independently by six intensive care experts working separately, and by the intelligent monitoring system PATRICIA. The results were recorded in charts such as that reproduced in Fig. 10. Finally, the results, decisions and interpretations of each expert were labeled A to F and

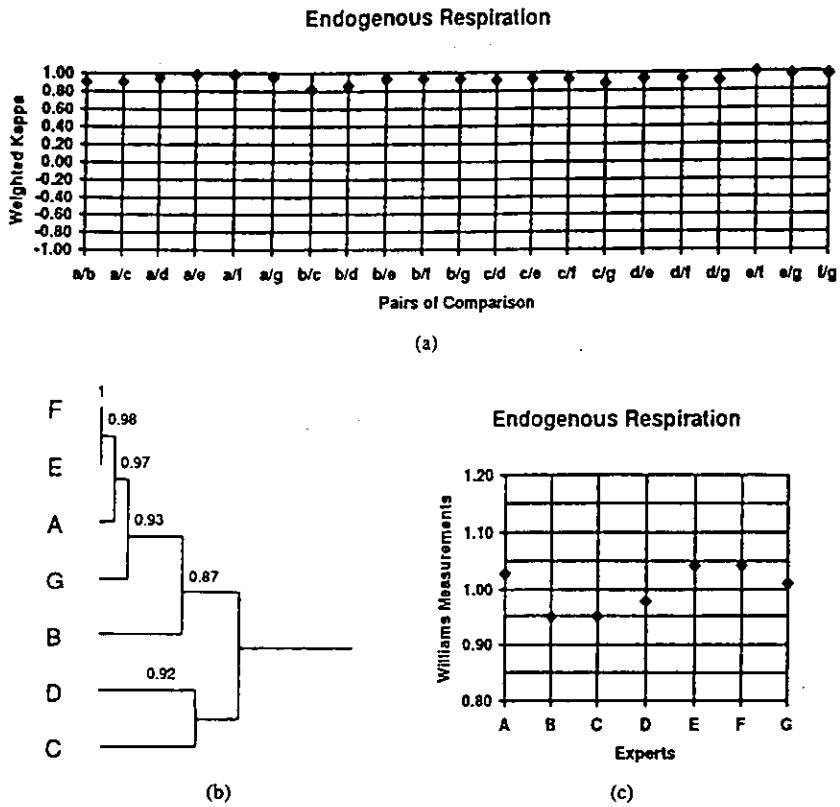


Fig. 11. Results for the endogenous respiration category using weighted kappa. (a) Tests between pairs of experts, (b) cluster analysis, and (c) Williams measurements.

TABLE I
RESULTS FOR THE DIAGNOSTIC CATEGORIES

	Oxygenation				Acid-base balance				Endogenous Respiration				Arterial Pressure				Heart Rate			
	%	%1	κ	κ_w	%	%1	κ	κ_w	%	%1	κ	κ_w	%	%1	κ	κ_w	%	%1	κ	κ_w
a/b	68	96	.56	.81	62	83	.56	.63	90	97	.82	.91	50	100	.30	.76	59	100	.35	.60
a/c	17	79	-.04	.19	62	79	.56	.63	83	100	.69	.91	47	80	.35	.63	66	86	.52	.38
a/d	28	90	.07	.52	83	90	.80	.86	90	100	.82	.95	72	100	.63	.88	52	97	.25	.48
a/e	79	100	.68	.88	83	93	.80	.87	97	100	.94	.98	60	100	.43	.81	62	100	.41	.67
a/f	83	100	.75	.90	62	90	.56	.72	97	100	.94	.98	63	97	.50	.78	66	97	.48	.53
a/g	28	76	.14	.58	86	90	.84	.86	90	100	.82	.96	43	100	.26	.77	59	100	.38	.67
b/c	17	66	.06	.20	53	87	.47	.63	83	93	.69	.82	20	70	.04	.52	34	90	.11	.49
b/d	28	76	.14	.40	60	80	.53	.62	79	97	.64	.86	55	100	.40	.80	45	90	.15	.28
b/e	62	100	.45	.83	57	83	.50	.64	93	97	.87	.93	77	100	.63	.87	45	97	.14	.43
b/f	69	93	.57	.76	57	77	.49	.64	93	97	.87	.93	70	100	.55	.83	45	90	.18	.24
b/g	45	86	.30	.76	53	80	.46	.60	93	97	.88	.93	60	93	.44	.74	34	90	.05	.23
c/d	63	87	.36	.33	67	83	.61	.73	87	100	.75	.92	41	83	.30	.73	45	83	.20	.20
c/e	20	80	.03	.24	77	83	.72	.72	87	100	.75	.93	33	73	.19	.60	62	86	.46	.47
c/f	17	73	-.04	.20	50	73	.43	.54	87	100	.75	.93	27	77	.13	.56	52	83	.31	.24
c/g	27	50	.20	.18	67	87	.61	.73	83	97	.69	.88	40	83	.27	.74	34	86	.06	.02
d/e	33	90	.16	.55	77	87	.73	.79	87	100	.75	.93	66	100	.53	.85	69	97	.47	.54
d/f	20	87	-.02	.45	53	77	.46	.60	87	100	.75	.93	48	97	.31	.73	76	93	.60	.54
d/g	30	53	.21	.29	70	83	.66	.78	80	100	.64	.91	55	100	.42	.83	69	100	.51	.70
e/f	60	100	.41	.77	60	87	.53	.74	100	100	1	1	67	100	.51	.82	72	97	.56	.65
e/g	30	70	.14	.59	87	97	.84	.93	93	100	.88	.97	53	97	.36	.76	48	100	.21	.53
f/g	30	83	.17	.58	60	83	.54	.67	93	100	.88	.97	63	93	.49	.76	62	97	.42	.60

those of PATRICIA were labeled G. The validation process took into account the following factors: symbolic processing of numerical parameters, individual diagnoses (e.g., heart rate), global diagnoses (e.g., acid-base balance) and finally, therapy prescriptions (e.g., ventilatory therapy)

Fig. 11 shows the results obtained for the cathegory "Endogenous Respiration" using the weighted kappa measurement and presented in a classical graphical way. However, due to space limitations, in the rest of this section results are presented through tables (Tables I–VI) in which "%" is the percentage

TABLE II
RESULTS FOR THE THERAPEUTIC CATEGORIES

	Oxygenatory				Ventilatory			
	%	%1	κ	κ_w	%	%1	κ	κ_w
a/b	62	97	.42	.57	61	93	.43	.64
a/c	57	97	.29	.43	57	97	.37	.66
a/d	66	100	.42	.61	52	97	.30	.65
a/e	67	97	.51	.66	60	93	.41	.62
a/f	76	100	.61	.75	64	93	.49	.68
a/g	60	97	.43	.64	60	97	.44	.72
b/c	55	97	.31	.47	61	96	.42	.68
b/d	61	93	.39	.40	70	96	.56	.77
b/e	69	100	.54	.78	75	96	.62	.78
b/f	61	93	.40	.44	81	100	.71	.89
b/g	59	93	.40	.57	68	100	.55	.83
c/d	55	93	.20	.17	76	97	.65	.78
c/e	50	93	.29	.40	83	97	.75	.81
c/f	59	97	.29	.40	71	93	.58	.68
c/g	57	93	.39	.46	80	97	.71	.82
d/e	48	93	.27	.36	83	97	.74	.83
d/f	75	100	.56	.69	70	96	.56	.68
d/g	41	93	.19	.34	76	100	.66	.87
e/f	59	93	.41	.50	79	93	.67	.74
e/g	67	93	.49	.66	87	97	.81	.86
f/g	62	93	.47	.54	82	96	.74	.85

TABLE III
WILLIAMS MEASUREMENT VALUES FOR THE DIAGNOSTIC CATEGORIES

	Oxygenation				Acid-base balance				Endogenous Respiration				Arterial Pressure				Heart Rate			
	%	%1	κ	κ_w	%	%1	κ	κ_w	%	%1	κ	κ_w	%	%1	κ	κ_w	%	%1	κ	κ_w
A	1.37	1.13	1.71	1.36	1.16	1.05	1.20	1.10	1.03	1.01	1.06	1.03	1.08	1.05	1.10	1.04	1.14	1.05	1.35	1.35
B	1.28	1.06	1.62	1.30	.82	.95	.78	.84	.99	.96	.99	.95	1.06	1.02	1.03	1.00	.73	.99	.42	.78
C	.58	.84	.30	.34	.93	.96	.92	.91	.93	.99	.86	.95	.58	.79	.48	.79	.85	.89	.81	.59
D	.77	.96	.53	.75	1.05	.98	1.06	1.04	.93	1.01	.87	.98	1.09	1.06	1.19	1.10	1.11	1.00	1.17	1.01
E	1.25	1.13	1.36	1.35	1.16	1.07	1.20	1.14	1.06	1.01	1.12	1.04	1.18	1.04	1.23	1.06	1.13	1.04	1.22	1.33
F	1.21	1.12	1.32	1.24	.82	.95	.78	.89	1.06	1.01	1.12	1.04	1.09	1.02	1.12	.99	1.19	.99	1.49	1.05
G	.71	.79	.70	.93	1.10	1.04	1.13	1.10	1.00	1.00	.99	1.01	.99	1.03	.97	1.03	.90	1.04	.79	1.02

TABLE IV
WILLIAMS MEASUREMENT VALUES FOR THE THERAPEUTIC CATEGORIES

	Oxigenation				Ventilatory			
	%	%1	κ	κ_w	%	%1	κ	κ_w
A	1.10	1.03	1.20	1.28	.77	.98	.63	.84
B	1.02	1.00	1.05	1.06	.96	1.01	.92	1.02
C	.89	.99	.68	.68	1.00	1.00	1.00	.97
D	.94	1.00	.81	.78	1.00	1.01	1.00	1.02
E	.99	.99	1.09	1.12	1.13	.99	1.22	1.04
F	1.12	1.00	1.23	1.10	1.07	.99	1.12	1.00
G	.94	.98	1.00	1.05	1.08	1.02	1.18	1.13

agreement, "%1" is the within-one percentage agreement, "κ" is the kappa measurement and "κ_w" is the weighted kappa measurement.

V. DISCUSSION

The results presented above show that, within the system's application domain; the conclusions arrived at by the intelligent monitoring system PATRICIA do not differ substantially from those arrived at by the experts involved in its validation. Looking at, for example the diagnostic category "endogenous respiration," percentage agreement is never less than 80% and within-one percentage agreement is practically 100%. Referring to the same category, kappa values (around 0.6) are adjusted upwards after carrying out weighting (weighted kappa around 0.8), which indicates that discrepancies observed are more a question of nuance rather than errors in interpretation. Again with respect to the same category, it can be seen that the cluster analysis based on within-one percentage agreements gives very little information, since virtually all experts are in agreement. Nevertheless, this same analysis carried out on percentage agreements and unweighted kappa measurement, permits the establishment of three groups: F-E-A, B-G and

TABLE V
CLUSTER ANALYSIS FOR THE DIAGNOSTIC CATEGORIES

	Oxygenation		Acid-base balance		Respiration		Arterial Pressure		Heart Rate	
%	f-a	.828	g-e	.867	f-e	1.000	e-b	.767	f-d	.759
	fa-e	.697	ge-a	.845	fe-a	.967	d-a	.724	fd-e	.707
	fae-b	.652	gea-d	.780	g-b	.931	f-eb	.683	e-a	.655
	d-c	.633	gead-c	.668	gb-fea	.915	g-feb	.600	fde-ca	.578
	g-faeb	.371	f-b	.567	d-c	.867	gfeb-da	.530	g-fdeca	.517
	gfaeb-dc	.257	geadc-fb	.549	gbfea-dc	.840	gfebda-c	.387	gfdeca-b	.401
%I	f-e	1.000	g-e	.967	g-f	1.000	f-e	1.000	g-e	1.000
	fe-a	1.000	ge-a	.914	gf-e	1.000	g-d	1.000	ge-a	1.000
	fea-b	.965	gea-f	.873	gfe-d	1.000	fe-b	1.000	gea-d	.974
	d-c	.867	c-b	.867	gfed-a	1.000	gd-a	1.000	gead-f	.948
	g-feab	.812	geaf-d	.820	gfeda-c	.998	gda-feb	.979	geadf-b	.914
	gfeab-dc	.644	geadfd-cb	.801	gfedac-b	.948	gdafeb-c	.770	geadfb-c	.866
κ	f-a	.752	g-e	.844	f-e	1.000	e-b	.630	f-d	.601
	fa-b	.567	ge-a	.819	fe-a	.937	d-a	.629	e-a	.524
	fab-e	.500	gea-d	.745	g-b	.878	f-eb	.526	fd-e	.514
	d-c	.357	gead-c	.614	gb-fea	.845	g-feb	.445	fde-ca	.374
	g-dc	.208	geadc-b	.496	d-c	.749	gfeb-da	.375	g-fdeca	.278
	gdc-fabe	.129	geadcb-f	.481	gbfea-dc	.707	gfebda-c	.263	gfdeca-b	.122
κ_w	f-a	.903	g-e	.925	f-e	1.000	d-a	.883	g-d	.698
	e-b	.833	ge-a	.866	fe-a	.985	e-b	.868	e-a	.667
	fa-eb	.806	gea-d	.819	g-fea	.966	f-eb	.826	f-ea	.589
	g-faeb	.627	gead-c	.700	gfea-b	.927	g-da	.801	gd-fea	.564
	gfaeb-d	.387	f-b	.640	d-c	.917	gda-feb	.768	c-b	.485
	gfaebd-c	.261	geadc-fb	.613	gfeab-dc	.875	gdafeb-c	.635	gdfea-cb	.268

TABLE VI
CLUSTER ANALYSIS FOR THE THERAPEUTIC CATEGORIES

	OXYGENATORY		VENTILATORY	
%	f-a	.759	g-e	.867
	fa-d	.703	ge-c	.817
	e-b	.690	f-b	.808
	g-eb	.626	gec-d	.776
	Fad-c	.564	gecd-fb	.707
	Geb-fadc	.547	gecdfb-a	.588
%I	f-d	1.000	g-d	1.000
	e-b	1.000	f-b	1.000
	fd-a	1.000	gd-fb	.973
	Fda-c	.957	c-a	.967
	Fdac-eb	.949	gdfb-e	.956
	g-fdaceb	.937	gdfbe-ca	.951
κ	f-a	.612	g-e	.806
	e-b	.539	ge-c	.728
	fa-d	.490	f-b	.706
	g-eb	.447	gec-d	.672
	Geb-fad	.352	gecd-fb	.566
	Gebfad-c	.294	gecdfb-a	.404
κ_w	e-b	.779	f-b	.891
	f-a	.752	g-d	.871
	fa-d	.648	gd-e	.847
	g-eb	.615	gde-c	.803
	Geb-fad	.463	gdec-fb	.724
	Gebfad-c	.370	gdecfb-a	.657

D-C. Implementing the corresponding cluster analysis using weighted kappa values, the results converge in that G approaches the first cluster F-E-A, whereas B moves away. An analysis of the results using Williams measurements produces conclusions conforming to those of the cluster analysis.

However, for other diagnostic/therapeutic categories (e.g., acid-base balance, arterial pressure, ventilatory therapy, etc.), the validation results are not so conclusive. In these cases it is certainly a question of discrepancies. In this respect when measuring agreement between pairs it is observed that discrepancies between PATRICIA and the experts is no greater than discrepancies between experts. In fact, information supplied by grouping techniques seems to indicate that PATRICIA tends to coincide more with the experts E, A, and F, rather than the rest.

Also noteworthy of mention are the analysis of results obtained in, one hand, the "heart rate" category and, in the other hand, in the "oxygenation" category when compared with the results of the "oxygenotherapy" category.

In the heart rate, a fact of note is that, although weighted kappa values are higher than kappa values and so maintain discrepancies in nuance, both measurements are, in fact, lower than those corresponding to the percentage agreement method.

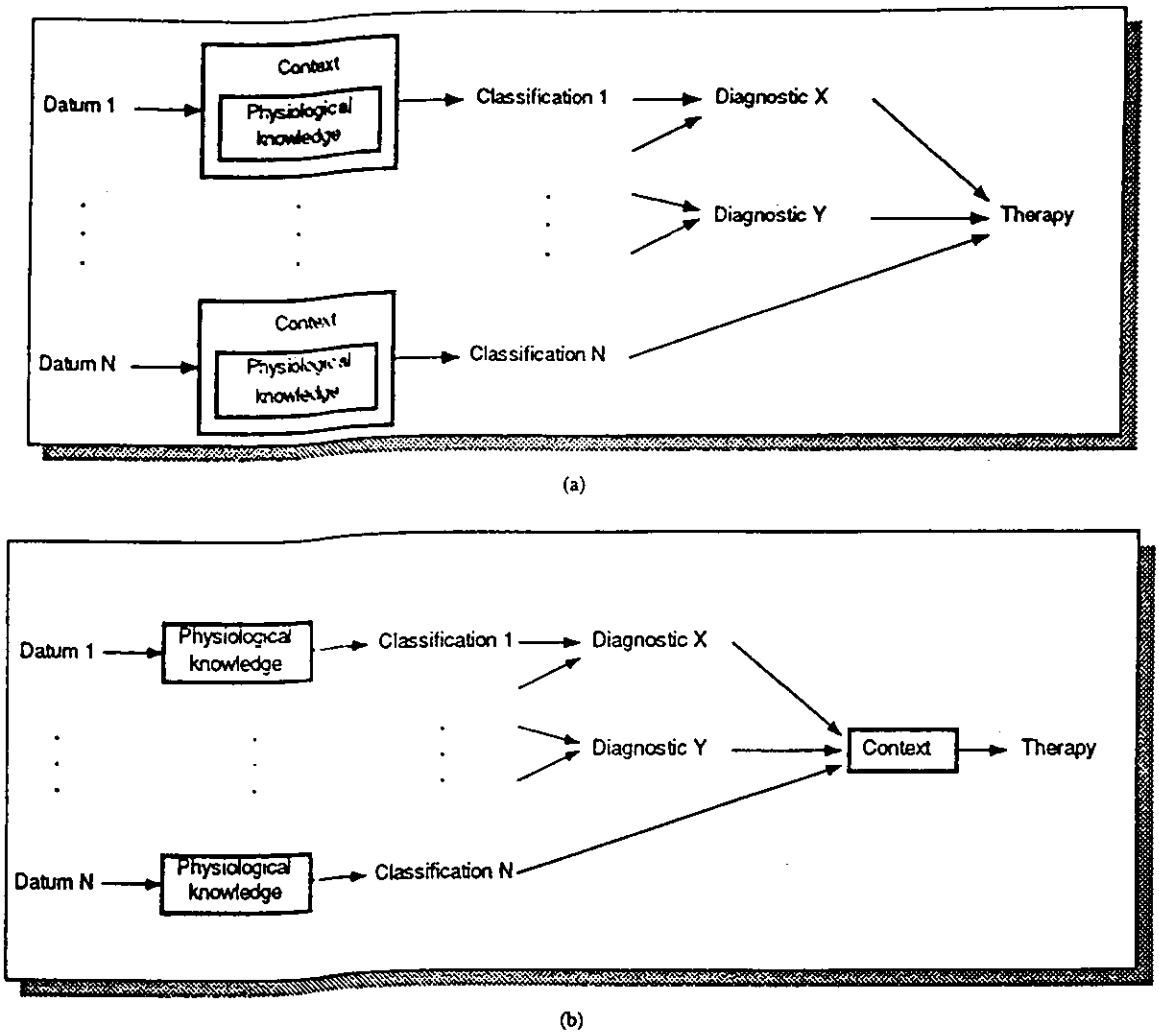


Fig. 12. (a) Process of diagnosis and therapy prescription for PATRICIA and (b) process of diagnosis and therapy prescription for the experts.

This situation however, can be explained by taking into account the previously mentioned limitations, with respect to sample characteristics of the kappa and weighted kappa methods. And so it appears that, for this particular category, the sample was not sufficiently representative.

A more detailed examination is required for the other two categories (oxygenation and oxygenotherapy). Why does PATRICIA differ so much from experts in the interpretation of the oxygenatory situation, and yet agree appreciably in the recommendation for oxygenatory therapy? The answer to this particular question may well be found in the way the experts and PATRICIA carry out the global inferential process, illustrated schematically in Fig. 12. It seems that the relative position of the contextual information used during the inferential process affects intermediate results; but not the final results (given that, that contextual information is taken into account anyway).

With respect to the methodology applied to evaluate the system, the authors consider that given the conditions and hypotheses described above, the use of quantitative techniques in a retrospective analysis context (in the evaluation of agreement between experts and in subsequent group agreement analysis) may play an important role in the validation of intelligent systems in critical domains.

To sum up:

- In the opinion of the authors, the acceptance of a given intelligent monitoring system owes much to the validation process.
- The choice of which validation technique to be used depends basically on the problem domain and on the characteristics of the intelligent system under consideration.
- The described methodology could be used in the validation of intelligent systems having similar characteristics of those applying to PATRICIA.

Finally, as far as PATRICIA is concerned, the authors believe that subsequent use of qualitative methods such as the "field test," could well be the next step in investigating the credibility of our intelligent system.

REFERENCES

[1] F. Hayes-Roth, D. A. Waterman, and D. B. Lenat, *Building Expert Systems*. Reading, MA: Addison-Wesley, 1983.

[2] R. O'Keefe and D. E. O'Leary, "Expert system verification and validation: A survey and tutorial," *Artif. Intell. Rev.*, vol. 7, pp. 3–42, 1993.

[3] G. Guida and G. Mauri, "Evaluating performance and quality of knowledge-based systems: Foundation and methodology," *IEEE Trans. Knowledge Data Eng.*, vol. 5, pp. 204–224, 1993.
[4] J. Geissman and R. D. Schultz, "Verification and validation of expert systems," *AI Expert*, vol. 3, no. 2, pp. 26–33, 1988.
[5] W. Adrion, M. Branstad, and J. Cherniavsky, "Validation, verification and testing of computer software," *ACM Comput. Surv.*, vol. 14, no. 2, pp. 159–192, 1982.
[6] F. A. Mora, G. Passariello, G. Carrault, and J. Le Pichon, "Intelligent patient monitoring and management systems: A review," *IEEE Eng. Med. Biol. Mag.*, vol. 12, pp. 23–33, 1993.
[7] K. Clarke, R. O'Moore, R. Smeets, J. Talmon, J. Brender, P. McNair, P. Nykanen, J. Grimson, and B. Barber, "A methodology for evaluation of knowledge-based systems in medicine," *Artif. Intell. Med.*, vol. 6, pp. 107–121, 1994.
[8] G. W. Williams, "Comparing the joint agreement of several raters with another rater," *Biometrics*, vol. 32, pp. 619–627, 1976.
[9] G. J. McLachlan, "Cluster analysis and related techniques in medical research," *Statist. Meth. Med. Res.*, vol. 1, pp. 27–48, 1992.
[10] C. Hernández, J. J. Sancho, M. A. Belmonte, C. Sierra, and F. Sanz, "Validation of the medical expert system RENOIR," *Comput. Biomed. Res.*, vol. 27, pp. 456–471, 1994.
[11] D. Ferrari, G. Serazzi, and A. Zeigner, "Clustering," in *Measurement and Tunning of Computer Systems*. Englewoods Cliffs: Prentice-Hall, 1983.
[12] R. O'Keefe, O. Balci, and E. Smith, "Validating expert system performance," *IEEE Expert*, pp. 81–87, Winter, 1987.
[13] B. Chandrasekaran, "On evaluating AI systems for medical diagnosis," *AI Mag.*, vol. 4, no. 2, pp. 34–37, 1983.
[14] A. Gonzalez and D. Dankel, "Verification and validation," in *The Engineering of Knowledge-Based Systems: Theory and Practice*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[15] J. R. Slagle, S. M. Finkelstein, L. A. Leung, and J. W. Warwick, "Monitor: An expert system that validates and interprets time-dependent partial data based on a cystic fibrosis home monitoring program," *IEEE Trans. Biomed. Eng.*, vol. 36, pp. 552–558, 1989.
[16] J. Cohen, "Weighted kappa: Nominal scale agreement with provision for scaled disagreement of partial credit," *Psych. Bull.*, vol. 70, no. 4, pp. 213–220, 1968.
[17] A. Alonso-Betanzos, L. D. Devoe, R. A. Castillo, V. Moret-Bonillo, C. Hernández-Sande, and N. S. Searle, "FOETOS in clinical practice: A retrospective analysis of its performance," *Artif. Intell. Med.*, vol. 1, no. 2, pp. 93–99, 1989.
[18] D. K. Donker, A. Hasman, and H. P. Van Geijin, "Kappa statistics: What does it say?," in *MEDINFO'92*, p. 901, 1992.
[19] V. Moret-Bonillo, A. Alonso-Betanzos, and C. Hernández-Sande, "Implementing cognitive procedures in diagnostic processes," in *Proc. 11th IEEE Eng. Med. Biol. Soc.*, 1989, pp. 1867–1868.
[20] V. Moret-Bonillo, A. Alonso-Betanzos, E. J. Truemper, E. García, and A. Pazos, "PATRICIA: An expert system that incorporates a patient-oriented approach for the management of ICU patients," in *Proc. 14th IEEE Eng. Med. Biol. Soc.*, 1992, pp. 876–877.
[21] V. Moret-Bonillo, A. Alonso-Betanzos, E. García, M. Cabrero, and B. Guijarro, "The PATRICIA project: A semantic-based methodology for intelligent monitoring in the ICU," *IEEE Eng. Med. Biol. Mag.*, vol. 12, pp. 59–68, 1993.
[22] V. Moret-Bonillo and A. Alonso-Betanzos, "Uncertainty based approach for symbolic classification of numeric variables in intensive care units," *J. Clinical Eng.*, vol. 15, no. 5, pp. 361–369, 1990.

Vicente Moret-Bonillo (M'88) was born in Valencia, Spain, in 1962. He graduated with the degree in physical chemistry from the University of Santiago de Compostela, Spain, in 1984, and the first postgraduate degree for research on control and monitoring of hemodynamic variables in 1985. He received the Doctorate degree (cum laude) in 1988 from the Department of Applied Physics, University of Santiago de Compostela, Spain, for work on the application of artificial intelligence techniques to respiratory treatment of patients dependent on mechanical ventilation.

From 1988 through 1990, he was a postdoctoral fellow in the Department of Biomedical Engineering Research, Medical College of Georgia, Augusta. He currently is a Profesor Titular de Universidad in the Department of Computer Science, University of La Coruña, Spain. His main current research areas are knowledge representation, application of knowledge engineering techniques to dynamic systems, and performance analysis of intelligent systems.

Dr. Moret-Bonillo is a member of various scientific societies, including the ACM.

Eduardo Mosqueira-Rey was born in La Coruña, Spain, in 1971. He graduated in computer science from the University of La Coruña, in 1994, where he received the first postgraduate degree in 1995 for work on the application of validation techniques to intelligent systems in critical domains. He is currently pursuing the Ph.D. degree in the Laboratory for Research and Development in Artificial Intelligence (LIDIA) on the subject of methodological approaches for the evaluation of intelligent monitoring systems.

His main current research areas are verification and validation of intelligent systems, intelligent monitoring systems, nonparametric and multivariate statistics applied to artificial intelligence, and analysis of techniques of uncertainty management.

Amparo Alonso-Betanzos (M'88) was born in Vigo, Spain, in 1961. She graduated with the degree in chemical engineering from the University of Santiago de Compostela, Spain, in 1984. In 1985 she joined the Department of Applied Physics, Santiago de Compostela, Spain, where she received the M.S. degree for work in monitoring and control of biomedical signals. In 1988, she received the Ph.D. (cum laude and premio extraordinario) degree for work in the area of medical expert systems.

From 1988 through 1990, she was a postdoctoral fellow in the Department of Biomedical Engineering Research, Medical College of Georgia, Augusta. She is currently an Associate Professor in the Department of Computer Science, University of La Coruña. Her main current research areas are expert systems, medical images, human motion analysis, and the integration of different artificial intelligence techniques in hybrid systems.

Dr. Alonso-Betanzos is a member of various scientific societies, including the ACM.

LNCS 1240: Biological and Artificial Computation: From Neuroscience to Technology.

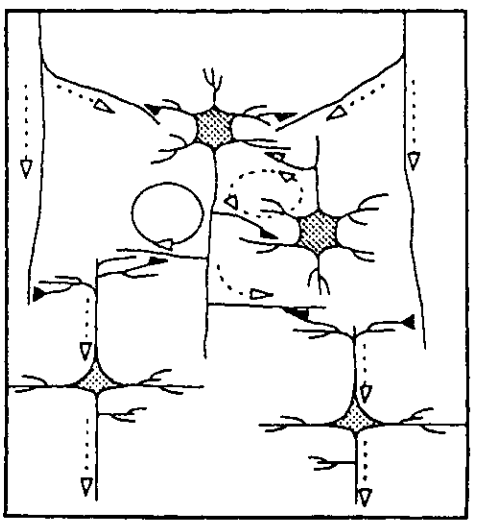
Alonso-Betanzos, A., Mosqueira-Rey, E. and Baldonedo del Río, B. "A Comparative Analysis of the Neonatal Prognosis Problem Using Artificial Neural Networks, Statistical Techniques and Certainty Management Techniques." in *Biological and Artificial Computation: From Neuroscience to Technology*, Lecture Notes in Computer Science, vol. 1240, Springer-Verlag, pp. 995-1004, 1997.

Lecture Notes in Computer Science 1240

José Mira Roberto Moreno-Díaz
Joan Cabestany (Eds.)

Biological and Artificial Computation: From Neuroscience to Technology

International Work-Conference on
Artificial and Natural Neural Networks, IWANN'97
Lanzarote, Canary Islands, Spain, June 1997
Proceedings



Springer

A Comparative Analysis of the Neonatal Prognosis Problem Using Artificial Neural Networks, Statistical Techniques and Certainty Management Techniques

A. Alonso-Betanzos, E. Mosqueira-Rey, B. Baldonedo del Río

LIDIA (Laboratorio de Investigación y Desarrollo en Inteligencia Artificial)
Department of Computer Science, University of A Coruña, 15071, A Coruña (Spain)
email: ciamparo@udc.es, eduardo@udc.es, belenb@udc.es

Abstract: One of the most popular methods for assessing fetal well-being is the nonstress test (NST). The expert system NST-EXPERT, performs a diagnosis of the nonstress test and formulates therapeutic plans, while taking into account different aspects of the maternal-fetal context, and incorporating these analysis into a model for predicting fetal outcome. The prognosis module actually implemented in the NST-EXPERT uses a mathematical model (based on the certainty factors of Shortliffe and Buchanan) that combines the NST diagnosis with the risk factors present in the maternal-fetal context to predict the fetal outcome. In this work we describe another different approaches followed based on artificial neural networks (ANN), statistical techniques (Bayes, logistic regression) and certainty management methods (Dempster-Shafer). The validation of the different models is performed through a formal methodology of validation using the tool SHIVA.

I. INTRODUCTION

The evaluation of intrauterine fetal condition is a standard of care in clinical obstetrics. Among the methods currently employed for antepartum fetal assessment, the nonstress test (NST), has become one of the most popular methods for assessing fetal well-being as it appears to be simple and reliable. The NST is based in the study of the Fetal Heart Rate (FHR) and the Uterine Activity (UA) signals taken into account the following parameters: (1) the fetal heart rate baseline (BFHR), (2) the variability of the BFHR (VAR), (3) the presence or absence of sufficient numbers of "good quality" accelerations, depending on gestational age (ACC), and (4) the presence or absence, quantity and type of the decelerations (DEC).

The expert system NST-EXPERT [1], performs a diagnosis of the nonstress test and formulates therapeutic plans, while taking into account different aspects of the maternal-fetal context, and incorporating these analysis into a model for predicting fetal outcome. In the current version, NST-EXPERT, is divided in four basic modules: (1) the *diagnostic module*, which contains a knowledge base (KB) so as to evaluate and interpret the results of the NST, and produce a final diagnosis of the test, (2) the *therapeutic module*, which works over the conclusions reached by the previous one and generates the therapeutic plan, (3) *The prediction module* of the system which is composed by a set of routines written in C language and a heuristic part which assigns the semantic labels (this module is fed with part of the data

interpreted by the diagnosis KB), and (4) the *user-system interface* developed in graphical environment making the interaction with the system more friendly.

The underlying model used in the prognostic module is based on Shortliffe and Buchanan's model and has been published elsewhere [2]. This model uses as evidences the NST diagnoses and the risk factors present in the maternal-fetal context. Analysis of the frequency and impact of the different high risk factors in our database yielded the following major risk factors: Intrauterine growth retardation (IUGR), Maternal hypertension (HPT), Diabetes (DBT) and Postdates (PDT). This factors accounted for more than 85% of all cases in the obstetric database and 93% of the poor outcomes not related to prematurity, congenital malformations or unavoidable accidents. The certainty values obtained from the data were refined by the experts in order to upgrade the capabilities of the prognostic module.

In this work we describe the different approaches followed while trying to solve the problem of the neonatal outcome prognosis. This approaches are based on artificial neural networks (ANN), statistical techniques (Bayes, logistic regression) and certainty management methods (Dempster-Shafer). The validation of the different models is performed through a formal methodology of validation using the tool SHIVA (an Spanish acronym that stands for Integrated and Heuristic System of Validation), a visual tool that allows to apply the methodology in a fast way.

II. Global Aspects of the Conexionist Approach

A very important step in the development of an artificial neural network is to choose the training data. We decide to use the database of more than 3000 cases that was used to calculate the certainty factors. In this cases, we only have the fetal outcome to perform the validation process. For this reason we acquire a new database (with 177 cases) in which we include the prognoses of three human experts and the expert system NST-Expert with the previous prognosis module.

In order to facilitate the feeding of the artificial neural network, a preprocessing phase was developed to convert the data into a binary value associating the value "1" with the presence of a risk factor or the abnormality of an NST parameter, and the value "0" with the absence of a risk factor or the normality of an NST parameter (the determination of normality is done through heuristic rules).The human experts prognoses were categorized in four mutually exclusive semantic labels (good, bad/slightly, bad/moderately, bad/quite) each one was identified with a value that represents presence (1) or absence (0). An example of the data can be seen in Fig. 1.

DBT	PDT	HPT	IUGR	FHRB	VAR	ACC	DEC	OUTC
0	1	0	0	1	1	0	0	0
0	0	0	0	0	0	0	1	0
0	1	0	0	1	1	0	0	1
1	0	0	1	0	0	0	0	0
1	0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0	1

Fig. 1. Example of the training data of the network.

In the different approaches to the problem we used network methodology known as multilayer perceptron that is characterized for [3]: (1) the model of each neuron in the network includes a *nonlinearity* at the output end (in our case the sigmoidal function), (2) the network contains one or more *hidden layers* (in our case only one), and (3) the network exhibits a high degree of *connectivity*. The reasons to use this topology are its simplicity and its efficiency proved in countless applications requiring static pattern classification.

The learning algorithm used is the back-propagation algorithm, in which we can distinguish two distinct passes: (1) a *forward pass* in which the synaptic weights remain unaltered and the input signal is propagated forward through the network and emerges at the output end as an output signal; and (2) a *backward pass*, in which an error signal generated at an output neuron is propagated through the network in order to adjust the network output to the desired output. The learning rule used is the *generalized delta rule* in which the learning process is modified through a *rate of learning* and a term called *momentum* that avoids the danger of instability.

The procedure used for stopping network training is the *Cross-validation method*. This method monitors the error on an independent set of data, called the test set, and stops training when this error begins to increase. This is considered to be the point of best generalization, avoiding the problems of overfitting.

III. Methodology of Validation

The methodology of validation used in the different models is represented in figure 2 and has been successfully applied to the expert system PATRICIA [4, 5].

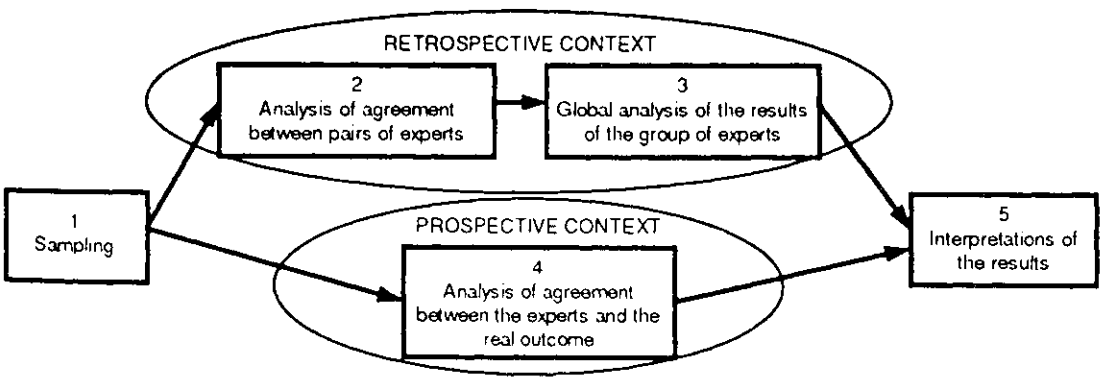


Fig. 2. Scheme of the different phases of the validation methodology

Phase 1: Sampling

For an accurate analysis of system performance it is essential to have available an adequate quantity of representative data relating to clinical cases. A part of the global cases captured was reserved to the validation process in order to ensure the generalization capabilities of the system.

Phase 2: Analysis of agreement between pairs of experts

For each diagnosis under consideration and for each pair of experts we build a contingency table used to compare the interpretations between to experts. These

tables should include the following aspects: (a) percentage agreement, (b) "within one" percentage agreement [6] (similar to the previous one but considering as "partial agreements" those diagnoses that differ only in one label in the semantic set), (c) kappa analysis (that quantifies the percentage agreement corrected for chance), and (d) weighted kappa analysis [7] (that weights disagreements according to their relative importance)

Phase 3: Global analysis of the results of the group of experts

Using the results obtained for pairs of experts, the use of Williams measurements [8] is recommended in order to determine to what extent agreement between system and group of experts deviates from agreement among experts. Cluster analysis [9] may also be useful in grouping expert and system results in classes.

Phase 4: Analysis of agreement between experts and fetal outcome.

Since we have a database with the prognoses of the experts and the fetal outcome, we can analyze the capacity of the experts in solving the given problem with the four accuracy ratios: sensitivity (true positive ratio), specificity (true negative ratio), false positive ratio and false negative ratio [10].

Phase 5: Interpretation of the results

We analyze the agreement between the system and the human experts (retrospective validation) and compare it with the capacity of the system in solving the problem in question (prospective validation).

IV. ANN MODELS FOR THE ANTENATAL PROGNOSIS

In this section we describe the different approaches followed when trying to solve the problem of the neonatal outcome prognosis through an artificial neural network model. For each of these approaches the training data, the topology of the network and the validation of the results are described. The common characteristics of all the models have been described in section II of this work.

1st Approach

In this approach we use as training data the database with more than 3000 cases in which we have the fetal outcome, and as test data the database with 177 cases in which we have both the fetal outcome and the prognoses of several domain experts. The network have 8 neurons in the input layer, 16 in a hidden layer and only one output neuron.

The problem that arises with this approach is that the outputs of the network are, most of the times, "good outcome" and we have only a few cases with output "bad outcome". The explanation of this problem can be the following: the training data only have two possible outcomes (good and bad) and these are the only possible outcomes of our network (it is not possible to obtain a range between good and bad). In the training data the number of good outcomes are far more frequent than the number of bad outcomes (corresponding to a representative distribution of the real data). Then the network tries to generalize all the cases to the most common outcome and the majority of the outputs are good outcomes. Moreover, we think that, perhaps, it is not a good idea to train the network with the real outcome, for example, a case

that presents some risk factors and some abnormal NST parameters is a good candidate to have a bad prognosis, but the subsequent medication of the patient may change her or his state and the fetal outcome may be good (representing a wrong correlation between risk factors and NST parameters with the fetal outcome). For these reasons we decide to use a different approach to the problem.

2nd Approach

In this approach we decide to develop a new network, but trained with the prognoses of the experts and not with the fetal outcome (since the opinions of the experts are categorized in four semantic labels we can develop a network with four possible outcomes, and not with only two). To do this we use the database in which we have both the fetal outcome and the prognoses of the experts and we set the 80% of the cases as training data (taking into account only the prognosis of the most prestigious expert) and the remaining 20% as test data (using all the expert's prognoses).

The topology of the network consists of 8 neurons in the input layer, 16 neurons in the hidden layer and 4 neurons in the output layer (that correspond to the 4 possible prognoses of the network: good, bad/slightly, bad/moderately and bad/quite). The prognoses of the network, the prognoses of the experts and the fetal outcome were passed to the validation tool SHIVA obtaining the following results in which ANN is the network; ES the expert system with the certainty values; and A, B and C are the human experts.

TABLE I
TESTS BETWEEN PAIRS OF EXPERTS (2^{nd} APPROACH)

	kappa	weighted kappa	percentage agreement	within-one percentage agreement
ES - A	0.146	0.404	0.543	0.914
ES - B	0.136	0.368	0.543	0.886
ES - C	0.287	0.584	0.600	0.943
ES - ANN	0.281	0.464	0.629	0.914
A - ANN	0.305	0.549	0.743	1.000
B - ANN	0.016	0.440	0.657	1.000
C - ANN	0.449	0.544	0.771	0.943
A - B	0.262	0.474	0.714	0.971
A - C	0.295	0.666	0.686	1.000
B - C	-0.063	0.311	0.543	0.971

In table I we show the results of the data belonging to the first phase of the retrospective validation that consists of an identification of the agreement between pairs of experts. In this table we can see that the results of all the experts are very similar (including the expert system with the certainty factors and the neural network). An important fact that can be noticed in this data is that kappa and weighted kappa values are very low. The explanation to this problems can be found in the definition of these measurements and in the characteristics of the sample. The sample used is representative of the most common cases found, in which we have more "good outcome" cases than "bad outcome" cases, which implies that the sample is unbalanced. Since kappa measurements correct agreements due to chance, if we have several values belonging to one type, this measurements interpret that exists a great factor of chance and the resulting value is low [11].

In tables II and III the information corresponding to the second phase in the retrospective validation, in which we use the information of the first phase, is shown. Using the Williams measurements (in which a value near one indicates that the prognoses of the expert under evaluation are similar to the prognoses of the other experts) we can see that all the results are near the unit. The only exceptions are the results of expert B in kappa measurements (is the most unbalanced sample) and the results of the expert system that are a little lower than the rest.

This situation is maintained after realizing the cluster analysis and grouping the experts according to the results of the agreement values of table I. Therefore we can see that the ANN is jointed, always at the first levels, with experts A and C, while in the following levels are jointed the expert system and the expert B (this one only in kappa measurements).

TABLE II
WILLIAMS MEASUREMENTS (2^{ND} APPROACH)

	kappa	weighted kappa	percentage agreement	within-one percentage agreement
ANN	1.483	1.067	1.157	1.018
ES	1.009	0.914	0.844	0.932
A	1.366	1.158	1.076	1.030
B	0.299	0.744	0.928	1.005
C	1.267	1.170	1.019	1.018

TABLE III
CLUSTER ANALYSIS (2^{ND} APPROACH)

kappa		weighted kappa		percentage agreement		within-one percentage agreement	
ANN - C	0.449	C - A	0.666	ANN - C	0.771	ANN - B	1.000
ANN.C - A	0.300	ANN - C.A	0.547	ANN.C - A	0.714	C - A	1.000
ANN.C.A - ES	0.215	ANN.C.A - ES	0.479	ANN.C.A - B	0.657	ANN.B - C.A	0.971
ANN.C.A.ES - B	0.128	ANN.C.A.ES - B	0.392	ANN.C.E.B - ES	0.561	ANN.B.C.A - ES	0.914

Then we can conclude that the results of all the experts are very similar, and that the discrepancies found between the ANN and the human experts are the same discrepancies that exist between the different experts. The fact that the results of the expert system are a little lower than the results of the other experts can be explained when performing the prospective validation (table IV) in which the prognoses of the experts are compared with the fetal outcome. The measurements used were the four accuracy ratios: sensitivity, specificity, false positive ratio and false negative ratio.

In the prospective validation is specially valued the ability to no diagnose as "good outcome" those cases in which the final outcome is bad (aspect more serious than diagnose as "bad outcome" a case in which the final outcome is good).

TABLE IV
PROSPECTIVE VALIDATION (2^{ND} APPROACH)

	Sensitivity (true positive ratio)	Specificity (true negative ratio)	False positive ratio	False negative ratio
ANN	0.871	0.75	0.25	0.129
ES	0.613	1	0	0.387
A	0.806	0.75	0.25	0.194
B	0.839	0.75	0.25	0.161
C	0.742	0.75	0.25	0.258

In this way we can see that all the experts (excluding the expert system) present true positive ratios higher than 0.7 and true negative ratios higher than 0.75. From this data we can conclude that the prognoses of the ANN are very similar to the

ognoses of the human experts, while the expert system is more pessimistic agnosing more "bad outcome" than the others. The advantage of this attitude is at the false positive ratio is zero and the expert system doesn't diagnose a bad case "good outcome".

3rd Approach

Although the results of the network were acceptable there were some input patterns for which the experts made different prognoses. To avoid the influence of the classification rules of the NST parameters (as normal or abnormal), we decided to train the network with the numerical values of the NST parameters. The results obtained with this approach do not differ substantially of the results obtained with the previous one, as we can see in the retrospective results of table V and the prospective results of table VI (in which the number of true positives is higher but the number of false positives is the same of the previous network).

TABLE V
AGREEMENT BETWEEN THE ANN AND THE EXPERTS (3RD APPROACH)

	kappa	weighted kappa	percentage agreement	within-one percentage agreement
ES - ANN	0.008	0.210	0.514	0.857
A - ANN	0.284	0.333	0.771	0.971
B - ANN	0.125	0.323	0.743	0.971
C - ANN	0.305	0.294	0.743	0.943

TABLE VI
PROSPECTIVE VALIDATION (3RD APPROACH)

	Sensitivity (true positive ratio)	Specificity (true negative ratio)	False positive ratio	False negative ratio
ANN	0.967	0.75	0.25	0.032

V. OTHER PREDICTION MODELS

As well as the different approaches made with the artificial neural networks, we tried to solve the problem using classical statistical techniques (Bayes theorem), new statistical techniques (logistic regression), and certainty management techniques (Dempster-Shafet model).

In order to apply Bayes model, we have to fulfill the following conditions: (1) the hypotheses must be mutually exclusive, (2) the set of hypotheses must be exhaustive and, (3) the evidences must be conditionally independent. The hypotheses are "good outcome" and "bad outcome" (that, obviously, fulfill the two first conditions). Nevertheless, when we use as evidences the risk factors and the NST parameters it is no clear that the third condition is fulfilled.

First we decided to suppose independence between all the evidences and apply Bayes theorem obtaining the prior probabilities of "bad outcome", $P(MS)$, and "good outcome", $P(\neg MS)$, and the conditional probabilities or likelihoods of the appearance of a given evidence when we know the fetal outcome. The results obtained with this approach are little different from the results of the experts, as we can see in the cluster analysis dendrogram (fig. 3) obtained through the validation process.

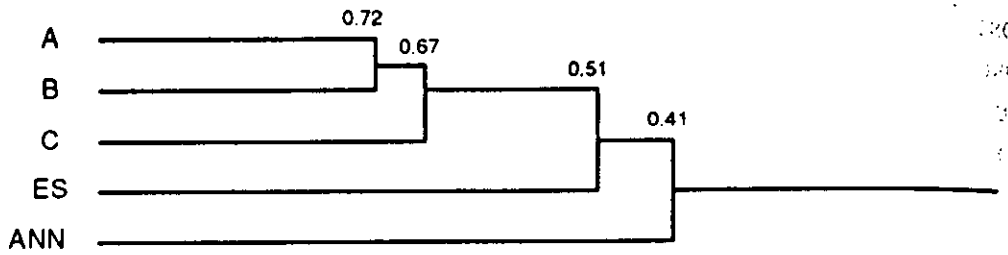


Fig. 3. Dendrogram of the cluster analysis for the results of Bayes theorem

Considering the prospective validation (table VII) the results of the true negative ratio and false positive ratio are acceptable, but the results of the true positives and the false negatives are not favorable.

TABLE VII
PROSPECTIVE VALIDATION (BAYES THEOREM)

	Sensitivity (true positive ratio)	Specificity (true negative ratio)	False positive ratio	False negative ratio
ANN	0.267	0.742	0.258	0.733

For this reason we decided to check if the evidences were really independent doing an independence analysis with the χ^2 of Pearson. The results seem to show that the risk factors are dependent between themselves and so are the NST parameters, but factors and parameters are independent. Then we decided to calculate the likelihoods of all the possible combinations of risk factors (2^4) and of all the possible combinations of parameters (2^4). Therefore, in a given case, the probability of "bad outcome" can be calculated using the product of the likelihoods obtained for the risk factors, in one hand, and for the NST parameters, in the other hand. This new approach cannot be applied because the lacking of some combinations of symptoms (i.e. in the database there was no cases that present all the risk factors or all the NST parameters abnormal).

Due to the problems found with Bayes theorem, we decided to use a more adequate statistical analysis, in which we do not need to presume independence between the evidences. Then we use the logistic regression model [12] in which a dichotomous variable (in this case the fetal outcome) depends on some variables called "independent variables" (risk factors and NST parameters). The equation that defines the model is equation 1, in which X_i represent the independent variables and P is the probability that the dependent variable takes a given value.

$$P = \frac{1}{1 + e^{-(\alpha + \sum \beta_i X_i)}} \quad (1)$$

In this equation there are some unknown terms (α y β_i) called the *regression coefficients* and which are necessary to infer from the data captured. The coefficient α is an independent term, while the terms β_i (there is one for each independent variable) represent the relation between each of the independent variables and the dependent variable (if it is positive the probability P is higher, if it is negative the probability P is lower and if it is 0 it means that the independent variable doesn't affect the dependent variable). The method applied to estimate the coefficients is the *maximum likelihood function*.

The problem found with this method is similar to the problem found in the first approach of the artificial neural networks, the results are a majority of "good outcome" except for some cases in which the outcome was bad. This means that the logistic regression tries to generalize the different cases presented, fitting them to the characteristics of the sample used to estimate the parameters of the model (a sample in which the number of good cases is high than the number of bad cases). With this method we cannot apply the solution used in the ANNs (using as training data the prognoses of the experts) because the dependent variable must be dichotomous.

In the certainty management methods, the results of the Shortliffe-Buchanan model have been published in [13]. Relating to the Dempster-Shafer model we decide to include in the *frame of discernment* two hypothesis: "good outcome" and "bad outcome" (that are mutually exclusive and exhaustive). The risk factors and the NST parameters represent the evidences of the model. Due to the special composition of the *frame of discernment* (with an hypothesis and its negation) it is easy to demonstrate that the resulting value of the application of the probability assignment function is equivalent to the value obtained with the combination of the different certainty factors of the Shortliffe-Buchanan model. For this reason the results obtained after the application of the Dempster-Shafer model are not shown because they are identical to the results of the certainty factors model.

V. Discussion

In this work we pretend to carry out a comparative analysis of the neonatal prognosis problem using artificial neural networks, statistical techniques and certainty management techniques.

In the ANNs the most accurate results were obtained in the networks trained with the prognoses of a prestigious human expert (second and third approach), while the results were worse when we trained the network with the fetal outcome (first approach). The explanation, already pointed, to this bad results may be that the network tries to approximate all the prognoses to the result "good outcome" because is the most frequent in the training data. One can think that this problem is easy to solve using a database with the same number of good and bad cases, but it is not so simple because this new database would not be representative of the real cases and because there are several similar cases, with a majority of good outcomes but with some bad outcomes, and if we eliminate some of this good outcomes the network can assume an erroneous correlation in the data diagnosing these cases as bad. The solution may be found by using more data to do the prognosis (more risk factors and contextual information) but it still has the problem of working with a real outcome that may have been affected by external factors unknown to the network (medication, clinical management, etc.). But the solution of training the data with human expert prognoses is not free from problems because we inherit the human errors and in some cases, attending to the false positive ratios, this errors may be very high.

With the certainty management models we can check that the Dempster-Shafer model is equivalent to the Shortliffe-Buchanan model in this case in particular, and the results are the same that have been published in [13].

In relation with the statistical measurements, the application of the theorem presented some problems associated with the constraint of independe the evidences (which is not easy to be sure to fulfill). The fact that the result little worse than the results obtained with the Shortliffe-Buchanan model indicate that the correction of the certainty factors made by the human experts crucial and the simple inference of conditional probabilities of a database may insufficient to obtain competitive results. Nevertheless the logistic regression model presents some advantages respect to the Bayes theorem, because it doesn't dema independent variables. The problem aroused with this model its the same proble found in the first approach of the ANN.

As a final conclusion we can say that there are several methods to implement prognosis of the fetal outcome, but the main advantage that presents the ANNs the rest is its flexibility, because there are several topologies and learning rules can be used without taking into account the constrains of the statistical methods the certainty management methods.

VI. ACKNOWLEDGMENTS

This work was funded in part by the Xunta de Galicia under the project XUG 10501B96

VII. REFERENCES

[1] A. Alonso-Betanzos, B. Guijarro-Berdiñas, V. Moret-Bonillo, and S. López, "The NST-EXP project: the need to evolve," *Artificial Intelligence in Medicine*, vol. 7, pp. 297-313, 1995.

[2] A. Alonso-Betanzos, V. Moret-Bonillo, L. Devoe, J. Searle, B. Banias, and E. Ramos, "Compu Antenatal Assessment: The 'NST-EXPERT' project," *Automedica*, vol. 14, pp. 3-22, 1992.

[3] S. Haykin, "Neural Networks: A Comprehensive Foundation," *Macmillan College Publi Company*, New York, 1994.

[4] V. Moret-Bonillo and E. Mosqueira, "A Methodology for Validating Intelligent Systems in Domains," in *Proc. of the 18th Annual Inter. Conf. of the IEEE EMBS*, 1996.

[5] V. Moret-Bonillo, E. Mosqueira, and A. Alonso-Betanzos, "Information Analysis and Validation Intelligent Monitoring Systems in Intensive Care Units" *IEEE Transactions on Biomedi Information Technology*, (submitted).

[6] J.R. Slagle, S.M. Finkelstein, L.A. Leung and J.W. Warwick, "Monitor: an expert system that valida and interprets time-dependent partial data based on a cystic fibrosis home monitoring program", *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 5, pp. 552-558, 1989.

[7] J. Cohen, "Weighted kappa: nominal scale agreement with provision for scaled disagreement of partial credit," *Psychological Bulletin*, vol. 70, pp. 107-121, 1968.

[8] G. W. Williams, "Comparing the joint agreement of several raters with another rater," *Biometrics*, vol. 32, pp. 619-627, 1992.

[9] G. J. McLachlan, "Cluster analysis and related techniques in medical research," *Statistical Methods in Medical Research*, vol.1, pp. 27-48, 1992.

[10] K. Adlassnig, and W. Scheithauer "Performance Evaluation of Medical Expert Systems Using ROC Curves," *Computers and Biomedical Research*, vol. 22, pp. 297-313, 1989.

[11] D. K. Donker, A. Hasman, and H. P. Van Geijin, "Kappa statistics: what does it say?," *MEDINFO'92*, pp. 901, 1992.

[12] D.G. Kleinbaum "Logistic Regression: A Self-Learning Text," *Springer-Verlag*, New York, 1992.

[13] A. Alonso-Betanzos, L.D. Devoe, R.A. Castillo, V. Moret-Bonillo, C. Hernández-Sande, and N.S. Searle, "FOETOS in clinical practice: A retrospective analysis of its performance," *Artificial Intelligence in Medicine*, vol. 1, pp. 93-99, 1989.